

# FACTORIZATION USING THE INFRASTRUCTURE OF BINARY QUADRATIC FORM CLASS GROUPS

Midshipman 1/C Stephen McMath

Advisers:

Professor W. David Joyner

Assistant Professor Frederick L. Crabbe, IV

Adviser endorsements:

_____	_____
_____	_____
Signature	Date

## **Abstract: Factorization Using the Infrastructure of Binary Quadratic Form Class Groups**

This research has analyzed algorithms for integer factorization based on continued fractions and binary quadratic forms, focusing on a runtime analysis of the algorithm and proving several valuable results about continued fractions.

Factorization is important for both practical and theoretical reasons. In secure digital communication, the RSA public key cryptosystem is often used. The security of this cryptosystem depends on the difficulty of factoring large integers. In number theory, factoring is of fundamental importance.

In 1975, Daniel Shanks used class group infrastructure to modify the Morrison-Brillhart algorithm and develop Square Forms Factorization, but he never published his work on this algorithm or provided a proof that it works. This research began by analyzing Square Forms Factorization, formalizing and proving the premises on which the algorithm is based and then analyzed two variations: one that uses a test of direction to perform a binary search and one that uses composition to divide the search between parallel processors.

Shanks' Square Forms Factorization, including a concept he called Fast Return, has been implemented in C and Magma and some experimental runtime analysis has been done. A parallel version in C has been implemented and tested.

## Acknowledgements

The thorough analysis of the contributors to this research would be an interesting historical investigation in its own right, but as space is limited, I will merely name a few. This must certainly start with my advisors. Throughout this research, Dr. David Joyner has allowed me to wander the theoretical realms while still keeping me grounded with his insistence on quality proofs of all important results. Dr. Frederick Crabbe, along with providing his computer expertise and understanding of the real world requirements for algorithms, has always found a way of adding a little humor to the mathematical analysis. Thanks must also go to Dr. Duncan Buell and Dr. Hugh Williams for providing me with the unpublished work by Daniel Shanks, along with providing their own research on the subject and aiding in the analysis of some of my work. The recent runtime analysis of Dr. Wagstaff and Jason Gower is greatly appreciated. Thanks must go to Dr. Carl Pomerance for his vision for the parallelization of these algorithms.

My family, although currently distant, has aided this project significantly. I must thank my parents for the curiosity they taught me and my brother LT Daniel McMath for starting me on this problem before I even knew the tools to begin to address it properly.

Lastly, I must thank my God. It is my faith in God that has given me confidence that there is order worth searching for and has given me the hope to keep me going when nothing else could have.

# Contents

	Abstract: Factorization Using the Infrastructure of Binary Quadratic Form Class Groups . . . . .	2
	Acknowledgements . . . . .	3
1	Introduction . . . . .	5
2	Background . . . . .	6
	The Problem and its Importance . . . . .	6
	Number Theory . . . . .	9
	Four Closely Related Tools . . . . .	12
3	History: From Morrison and Brillhart to Present . . . . .	16
4	Square Forms Factorization . . . . .	18
	Original Algorithm and Variations . . . . .	18
	Parallelization . . . . .	22
	Factorization using a Test of Direction . . . . .	23
	Secure and Insecure Numbers . . . . .	25
	Bibliography . . . . .	28
	Index . . . . .	31
<b>A</b>	<b>Daniel Shanks' Square Forms Factorization</b>	<b>32</b>
1	Introduction . . . . .	32
2	Continued Fractions . . . . .	33
3	From Morrison-Brillhart to Shanks . . . . .	41
4	Quadratic Forms . . . . .	43
5	Ideals . . . . .	48
6	Lattices . . . . .	58
7	The Generalized Distance Formula . . . . .	64
8	Square Forms Factorization (SQUFOF) . . . . .	68
<b>B</b>	<b>Class Group Related Programs</b>	<b>72</b>
<b>C</b>	<b>Parallel Programs Using MPI</b>	<b>81</b>

# 1 Introduction

The problem of distinguishing prime numbers from composite numbers and of resolving the latter into their prime factors is known to be one of the most important and useful in arithmetic. It has engaged the industry and wisdom of ancient and modern geometers ... the dignity of the science itself seems to require that every possible means be explored for the solution of a problem so elegant and so celebrated.

C. F. Gauss [7]

Factorization is important for both practical and theoretical reasons. In secure digital communication, the RSA public key cryptosystem is often used. The security of this cryptosystem depends on the difficulty of factoring large integers. In number theory, factoring is of fundamental importance.

The algorithms for factoring larger and larger integers quickly have developed significantly over the years. There are many ways of doing this, ranging from trial division to the number field sieve. This research focuses on a specific underdeveloped algorithm introduced by Daniel Shank 1975, Square Forms Factorization, but also analyzes two new variations, one due to a conjecture by Pomerance and one introduced by this author. The overall goal of this research was to analyze algorithms for integer factorization based on the use of continued fractions and quadratic forms, focusing on proving mathematical results in these areas and determining runtime. We proposed several preliminary sub-goals to this end:

1. Analyze the conditions for which Square Forms Factorization provides a factorization.
2. Analyze the connection between continued fractions and quadratic forms and provide related proofs.
3. Analyze a test of direction.
4. Produce a computer implementation of these algorithms.

The first sub-goal was to analyze the conditions for which Square Forms Factorization provides a factorization. This information is important to cryptology, as the security of some public key cryptosystems is dependent on the difficulty of factorization. The conditions for which the algorithms must provide a factorization have been determined. However, a sufficient set of conditions for which the algorithms will not work has not yet been determined. In addition to the original goal, this research is currently analyzing which numbers Square Forms Factorization may factor significantly faster or slower than others, which is just as important since numbers that factor extremely slowly are almost as secure as numbers that the algorithm doesn't factor at all. It is possible to determine sufficient conditions for which a number factors quickly but necessary conditions remain elusive. Specifically, if  $N = pq$  where the ratio  $p/q$  is close to a ratio of perfect squares,  $N$  will factor unusually fast. Conjecture 2 of §4 explains this in more detail.

The second sub-goal was to analyze the connection between continued fractions and quadratic forms, specifically intending to analyze the work done by Shanks. This research has taken significantly longer than planned for in the original timeline. Much of the existing information on these theories was scattered and disorganized, and many of the papers either lacked proofs or contained errors, so it was valuable to organize this information into a form that could be useful. The most important achievement in this was the development and proof of a formula relating infrastructure distance with composition of quadratic forms.

The third sub-goal was to analyze a test of direction useful to a new algorithm being developed. Very little progress has been made on this so far. Although the other proofs on quadratic forms and continued fractions have provided significant insight, they have provided very little insight into the test of direction.

The fourth of the original sub-goals was to produce a computer implementation of these algorithms. Shanks' Square Forms Factorization has been implemented in both C and Magma, including libraries of functions concerning the operations essential to quadratic forms and continued fractions. This source code will be made available in a distributable form. The Magma version is included in Appendix B.

Concerning runtime, Jason Gower [8] has recently analyzed the runtime of SQUFOF for his Ph.D. thesis at Purdue University, so this investigation will not continue further in that direction. Included in this was an analysis of multipliers, a concept that will be used in these implementations. However, it has been conjectured by Pomerance that as SQUFOF is easy to parallelize, a parallel implementation may be competitive, so a new goal has been set of analyzing the parallel implementation of SQUFOF.

A parallel version of SQUFOF has been developed and has been implemented and tested in C. This code is included in Appendix C. No comparison of the runtime of this algorithm has been made yet.

§2 provides some background to the problem of factorization, basic number theory, and some of the tools related to Square Forms Factorization. §3 describes how Square Forms Factorization developed from the existing theory. §4 describes Square Forms Factorization, including several variations of the algorithm, and describes some significance the algorithm may have for cryptology.

## 2 Background

### The Problem and its Importance

There are several different kinds of factorization; this research will focus on integer factorization. Consider an integer  $N$ . Factorization is the process of finding integers  $p$  and  $q$  greater than 1 such that  $N = pq$ . Complete factorization would require repeating this process for both  $p$  and  $q$  until all of the remaining factors are prime. However, if an algorithm can be developed to quickly factor  $N$  into  $p$  and  $q$ , the same algorithm can be used over again on  $p$  and  $q$ . For example, it is easy to see that  $105 = 5 \cdot 21$  and then repeat to factor 21. From here, you would see that since  $21 = 3 \cdot 7$ ,  $105 = 5 \cdot 3 \cdot 7$ . Although in this simple case, the complete factorization is easy to find, this task becomes much harder for large numbers.

## Number Theoretic Applications

In number theory, the factors of a number dictate many of the characteristics of the number. For example, Euler's  $\phi$  function, which tells how many numbers less than  $N$  are relatively prime to  $N$ , can be directly calculated from the complete factorization. For an integer that is the product of distinct odd primes,  $\phi(N)$  may be calculated by multiplying each of the factors minus 1.

**Example 1**  $105 = 5 \cdot 3 \cdot 7$ . Therefore  $\phi(105) = (5 - 1)(3 - 1)(7 - 1) = 48$ , so there are 48 integers less than 105 that are relatively prime to 105.

Also, determining whether a number is a *quadratic residue* (i.e. the square of another number modulo  $N$ ), can be determined directly using Gauss's Quadratic Reciprocity Law if the complete factorization of  $N$  is known.

**Example 2**  $19^2 = 361 = 46 + 3 \cdot 105$ , so that 46 is a quadratic residue modulo 105 with square root 19. One would represent this as  $19^2 \equiv 46 \pmod{105}$ . With the complete factorization, we can use Gauss's Quadratic Reciprocity Law to analyze 46 modulo 3, 5, and 7 to determine whether or not 46 is a quadratic residue modulo 105 without actually having to find its square root first [7].

Therefore, factoring large numbers has been a focus of research for a variety of theoretical reasons.

## Cryptographic Applications

One practical application of factorization is public key *cryptography*. The idea of public key cryptography is that it is possible to keep a communication secret without having to keep the key secret. The message is encrypted by one key, which is made public, and is decrypted by another key, which is kept secret. In the most prevalent public key encryption system, *RSA*<sup>1</sup> [24], the cryptographer chooses a number that is the product of two large prime<sup>2</sup> numbers  $p$  and  $q$ :  $N = pq$ . Then an exponent  $e$  is chosen such that  $e$  is relatively prime to  $(p - 1)(q - 1)$ . Although there are several variations, in the normal public key version,  $N$  and  $e$  are made public. The user of the RSA system then privately calculates  $d$ , the inverse of  $e$  modulo  $(p - 1)(q - 1)$ . Anyone is able to encrypt something to him by raising blocks of the message to the power  $e$ , modulo  $N$ :  $c \equiv m^e \pmod{N}$ , where  $m$  is the original message and  $c$  is the encrypted message. Then, the recipient is able to decrypt by evaluating  $m \equiv c^d \pmod{N}$  [24].

### Example 3

$$p = 4327, \quad q = 1009, \quad N = pq = 4365943$$

---

<sup>1</sup>RSA is named after Rivest, Shamir, and Adleman. It was earlier developed by Clifford Cooks of GCHQ, but this was only recently declassified [6].

<sup>2</sup>Usually, these are just numbers that pass several primality tests and thus have a high probability of being prime, called pseudo-primes. Very rarely, one of them will not be, but this is rare enough to not cause significant problems.

$$e = 2005$$

and from this the cryptographer would use  $p, q$  and  $e$  to calculate  $d = 865597$ . He would make  $N$  and  $e$  public. Suppose someone wants to send him the message 809.

The sender would evaluate  $809^{2005} \pmod{N}$  as 1591327. This would be sent over the internet.

The receiving computer would then evaluate  $1591327^{865597} \pmod{N}$  as 809. Since this is the only computer with access to  $d$ , anyone else intercepting the message would have great difficulty in determining the message

Another application of public key cryptography is for signatures. If the sender uses his private key to encrypt a message, the receiver may use the public key to decrypt the message and be certain of who the message originated from and that no one along the way has modified it. Often this is combined with a second layer of encryption to prevent anyone else from also being able to use the public key to read the message.

As long as someone intercepting a message is unable to factor  $N$ , it is usually impossible to obtain  $d$ , so that the message cannot be broken. The security of RSA and its variations depends highly on whether or not  $N$  can be factored [31]. Although extremely fast factorization would be a threat to these systems, the advances in number theory produced by faster factorization would likely provide a number of alternative secure systems. Also, in addition to the potential for alternative secure systems, there is the strong possibility that fast factorization algorithms will work better for some numbers than others. If there are classes of numbers that a faster factorization algorithm does not work on, this would enable designers of the algorithm to increase their security by relying more on these numbers. Regardless of whether or not the algorithm works for all numbers or provides alternative systems, for security purposes it is necessary to understand the strengths and weaknesses of the algorithm.

## Runtime Analysis

Up to this point we have referred to the speed of factorization in general terms, but there are several different ways to classify the speed of an algorithm. Let  $N$  be the number to factor. Let  $n$  be the number of bits in  $N$ ,  $n = \log_2 N$ . An algorithm's run time is called *exponential* if it increases exponentially with the size of the input, in this case  $n$ . *Linear* refers to an algorithm where the time increases proportionally to the number of bits<sup>3</sup>. *Polynomial* refers to an algorithm for which the time required is some polynomial function of  $n$ . Thus, linear time is a special case of polynomial time. There are some algorithms that fall in between polynomial and exponential time and are referred to as sub-exponential.

Note that different algorithms may be faster for different sizes of numbers or even for different systems. This is why the runtime is analyzed in terms of growth. For small values of  $n$ , linear and exponential may be fairly close or linear may even be faster, but the runtime

---

<sup>3</sup>Since  $n = \log_2 N$ , so that such a runtime is logarithmic in  $N$ , this is often referred to as logarithmic, resulting in a certain amount of confusion. Note that both ways of expressing runtime will be used in this paper.

of an exponential algorithm will grow faster with the size of  $n$ , so that for sufficiently large  $n$ , an exponential algorithm will be slower than a sub-exponential, which will be faster than a polynomial, which will be faster than a linear.

The runtimes for sub-exponential algorithms are described by complicated formulas. Define

$$L(\alpha, c) = \exp(cn^\alpha(\log n)^{1-\alpha})$$

If  $\alpha = 1$ , this is an exponential algorithm. If  $\alpha = 0$ , this is a polynomial algorithm. For values of  $\alpha$  in between 0 and 1, the algorithms are sub-exponential.

In terms of asymptotic runtime, the best general purpose factorization algorithm is the *general number field sieve*, with a runtime<sup>4</sup> of  $L(1/3, \frac{4}{3^{2/3}} + o(1))$  [14]. Two other algorithms are currently in common use, the *elliptic curve method*, which has a runtime of  $L(1/2, 1+o(1))$  [16] and the *multiple polynomial quadratic sieve*, which has a runtime of  $L(1/2, 1)$  [20]. Although each of these algorithms is slower on average, they are each faster for some types of integers, so that in combination they tend to be faster than the number field sieve.

Another less commonly considered characteristic of a factorization algorithm is the efficiency with which it can use multiple processors working simultaneously. The current trend in ultra-fast computing is to use a large number of inexpensive processors instead of a single expensive processor. Given this trend, it is important that a good algorithm be able to use a multiple processor system efficiently. Dividing a single task between multiple processors is called *parallelization*. With perfect efficiency, factoring a number with 10 processors should take on average one tenth the time required for factorization on a single processor, but no algorithm can ever quite attain perfect efficiency.

## Number Theory

Mathematics is the Queen of the sciences, and arithmetic the Queen of mathematics.

C. F. Gauss [25]

This section provides a quick overview of basic number theory concepts and notation that will be used throughout this report. Those already familiar with number theory are welcome to skip this section, as none of this is original and may be found in any standard number theory text [9].

Number theory analyzes sets of numbers. If an element  $e$  is in a set  $S$ , this is written  $e \in S$ . If an element  $e$  is not in a set  $S$ , this is written  $e \notin S$ . If all the elements in set  $S_1$  are in set  $S_2$ , then  $S_1$  is contained in set  $S_2$  and this is written  $S_1 \subset S_2$ . If  $S_1 \subset S_2$  and  $S_2 \subset S_1$ , then  $S_1 = S_2$ .

The set of all real numbers is denoted  $\mathbb{R}$ . Although this set will be important, it is primarily several special subsets of the real numbers that this research will focus on:

---

<sup>4</sup>The expression  $o(1)$  is *little o* notation. It converges to 0 as  $n$  goes to infinity. An understanding of little  $o$  notation is not important for understanding this report.

The first subset of the real numbers is the set of integers, represented as  $\mathbb{Z}$ .

$$\mathbb{Z} = \{\dots - 3, -2, -1, 0, 1, 2, 3\dots\}.$$

The natural numbers are the integers  $\geq 1$ . The operations on the integers are the standard addition and multiplication. Addition has the usual four basic properties: associativity, commutativity, identity, and inverses.

Multiplication has three basic properties: associativity, commutativity, and identity.

One other property, distributivity, involves both addition and multiplication:

$$a \cdot (b + c) = ab + ac.$$

There are several other relations that are important in  $\mathbb{Z}$ . A natural number  $a$  divides an integer  $b$  if there exists an integer  $k$  such that  $b = ka$  and this is written  $a \mid b$ . If  $a$  does not divide  $b$ , this is written  $a \nmid b$ . A natural number  $> 1$  is *prime* if it may only be divided by itself and 1. There are several basic properties of this relation:

If  $a \mid b$  and  $b \mid a$ , then  $a = b$ .

If  $a \mid b$  and  $a \mid c$ , then  $a \mid (b + c)$ .

If  $a \mid b$ , then  $a \mid bc$ .

If  $a$  is prime and  $a \mid bc$ , then  $a \mid b$  or  $a \mid c$ .

If for some integer  $n$  and a prime  $p$ ,  $p^n \mid b$ , but  $p^{n+1} \nmid b$ , this is written  $p^n \parallel b$ .

$c$  is the *greatest common divisor* of  $a$  and  $b$  if  $c > 0$ ,  $c \mid a$ ,  $c \mid b$ , and for any integer  $d$  such that  $d \mid a$  and  $d \mid b$ ,  $d \mid c$ . This is written  $c = \gcd(a, b)$  or sometimes (although not in this report) as merely  $(a, b)$ .

Two integers  $a$  and  $b$  are *relatively prime* if  $\gcd(a, b) = 1$ , that is, if  $a$  and  $b$  have no factor in common.

If  $p, m, n$  are integers with  $p$  prime such that  $p^m \parallel a$  and  $p^n \parallel b$ , then  $p^{\min(m, n)} \parallel \gcd(a, b)$ .

If  $c = \gcd(a, b)$ , there exist integers  $x, y \in \mathbb{Z}$  such that  $ax + by = c$ .  $\gcd(a, b)$ , along with these two integers, may be efficiently calculated using Euclid's extended algorithm.

$c$  is the *least common multiple* of  $a$  and  $b$  if  $c > 0$ ,  $a \mid c$ ,  $b \mid c$ , and for any integer  $d$  such that  $a \mid d$  and  $b \mid d$ ,  $c \mid d$ . This is written  $c = \text{lcm}(a, b)$  or sometimes merely as  $\{a, b\}$ . If  $p, m, n$  are integers with  $p$  prime such that  $p^m \parallel a$  and  $p^n \parallel b$ , then  $p^{\max(m, n)} \parallel \text{lcm}(a, b)$ .

The gcd and lcm are related by the formula  $\gcd(a, b) \cdot \text{lcm}(a, b) = ab$ .

If  $c \mid (a - b)$ , then we say that  $a$  is *congruent* to  $b$  modulo  $c$ , written  $a \equiv b \pmod{c}$ . This relation has several basic properties:

If  $a \equiv b \pmod{n}$ , then  $b \equiv a \pmod{n}$ .

If  $a \equiv b \pmod{n}$  and  $b \equiv c \pmod{n}$ , then  $a \equiv c \pmod{n}$ .

If  $a \equiv b \pmod{n}$  and  $c \equiv d \pmod{n}$ , then  $a + b \equiv c + d \pmod{n}$  and  $ac \equiv bd \pmod{n}$ .

By using Euclid's extended algorithm to compute greatest common divisors, for  $a$  relatively prime to  $b$  it is possible to calculate  $a^{-1} \pmod{b}$ . Thus, division modulo  $b$  is defined as reducing to least terms and then multiplying by the inverse.

One important type of number is a *quadratic residue*. The integer  $a$  is a quadratic residue of integer base  $b$  if there exists some integer  $x$  such that  $x^2 \equiv a \pmod{b}$ . For example, 2 is a quadratic residue of 7 because  $3^2 = 9 \equiv 2 \pmod{7}$ . The symbol used is  $\left(\frac{a}{b}\right)$ .  $\left(\frac{a}{b}\right) = 1$  if

$x^2 \equiv a \pmod{b}$  has a solution,  $-1$  if it does not, and  $0$  if  $a$  and  $b$  are not relatively prime. If  $a$  is a quadratic residue of  $b$  and  $a$  is a quadratic residue of  $c$  and  $b$  and  $c$  are relatively prime, then  $a$  is a quadratic residue of  $bc$ . Gauss's Quadratic Reciprocity Law provides a fast way of determining quadratic residues modulo a base whose prime factorization is known:

**Theorem 1** *For  $p$  and  $q$  distinct primes:*

$$\begin{aligned}\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) &= (-1)^{(p-1)(q-1)/4} \\ \left(\frac{2}{p}\right) &= (-1)^{(p^2-1)/8} \\ \left(\frac{-1}{p}\right) &= (-1)^{(p-1)/2}\end{aligned}$$

There are many other well-known properties concerning congruences. For example, “Fermat’s little theorem”: for  $p$  prime and  $a$  an integer if  $p \nmid a$ , then  $a^{p-1} \equiv 1 \pmod{p}$ . This is often used to test whether not an integer  $N$  is prime. (If for some  $a < N$ ,  $a^{N-1} \not\equiv 1 \pmod{N}$ , then  $N$  is not prime. However, there are infinitely many composite numbers that will appear prime for any  $a$  such that  $\gcd(a, N) = 1$ , so the converse is not necessarily true [1].)

In  $\mathbb{Z}$ , inverses are not defined for multiplication. However, this question brings us to the set of rational numbers, represented as  $\mathbb{Q}$ .

$$\mathbb{Q} = \{a/b : a, b \in \mathbb{Z}, b \neq 0\}$$

All of the properties for addition and multiplication in  $\mathbb{Z}$  still apply, in addition to one more:

If  $a \in \mathbb{Q}$  and  $a \neq 0$ , then there exists a unique integer  $b$  such that  $a \cdot b = 1$ . This is written as  $a^{-1}$  or  $1/a$ .

Let  $\alpha$  denote a real or complex number not belonging to  $\mathbb{Q}$ . Denote the set of all finite linear combinations with integer coefficients of all the non-negative powers of  $\alpha$  by,

$$\mathbb{Z}[\alpha] = \{a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + \dots : a_i \in \mathbb{Z}\}.$$

For example, if  $\alpha = \sqrt{2}$ , then  $\mathbb{Z}[\sqrt{2}]$  contains  $5 + 3\sqrt{2}$  and  $4 - 2\sqrt{2}$ , but does not contain  $\frac{4}{\sqrt{2}}$ .

Alternately, the new set could be the set of all linear combinations of all the powers of  $\alpha$ . That is,

$$\mathbb{Q}(\alpha) = \{\dots a_{-2}\alpha^{-2} + a_{-1}\alpha^{-1} + a_0 + a_1\alpha + a_2\alpha^2 + \dots : a_i \in \mathbb{Q}\}$$

For example,  $\mathbb{Q}(\sqrt{2})$  contains both  $3 + \sqrt{2}$  and  $4 + \frac{2}{\sqrt{2}}$ , but does not contain  $\frac{1}{1+\sqrt{2}}$ .

For an element  $\zeta$  in either of the sets  $\mathbb{Z}[\sqrt{N}]$  and  $\mathbb{Q}(\sqrt{N})$ ,  $\bar{\zeta}$  refers to the *conjugate* of  $\zeta$ , which is found by changing the sign of the algebraic part. For example, if  $\zeta = 1 + \sqrt{3}$ , then  $\bar{\zeta} = 1 - \sqrt{3}$ .

The *norm* of  $\zeta$  is  $\mathcal{N}(\zeta) = \zeta\bar{\zeta}$ .

## Four Closely Related Tools

This section introduces four areas of mathematics that this research has focused on: continued fractions, quadratic forms, ideals, and lattices. These areas are closely related and therefore have the same property of being periodic. See Appendix A for a more detailed analysis of each of these topics, along with related proofs and an analysis of their connections.

1. A *continued fraction* is a tool originally used for rational approximation. Given a number  $\alpha$ , the goal is to represent  $\alpha$  in the form

$$\alpha = b_0 + \frac{1}{b_1 + \frac{1}{b_2 + \dots}}$$

for integers  $b_i$ . This is often abbreviated as  $[b_0, b_1, b_2, \dots]$ . The expressions found by truncating this,  $b_0, [b_0, b_1], [b_0, b_1, b_2], \dots$  are called the *convergents*. These simplify down to rational numbers  $A_n/B_n$ .

The sequence of  $b_i$ 's is found by the recursive formulas

$$x_0 = \alpha, \quad b_0 = \lfloor x_0 \rfloor \tag{1}$$

$$\forall i \geq 1 \quad x_i = \frac{1}{x_{i-1} - b_{i-1}}, \quad b_i = \lfloor x_i \rfloor \tag{2}$$

Note that this formula is derived by solving the equation  $x_{i-1} = b_{i-1} + \frac{1}{x_i}$  for  $x_i$ .  $\lfloor x \rfloor$  refers to the *floor* of  $x$ , the greatest integer less than or equal to  $x$ .

If  $x_0 = \sqrt{N}$  for  $N$  a non-square integer (that is, not the square of another integer), then this sequence may be calculated efficiently. In particular, each  $x_i$  reduces to the form  $\frac{\sqrt{N}+P}{Q}$  with  $P$  and  $Q$  integers. The integer  $P$  in the numerator is called the *residue*. The denominator  $Q$  is called a *pseudo-square*. There are formulas to calculate these recursively, after the first several steps, without doing arithmetic on any integer larger than  $\sqrt{N}$ .

The rational approximation produced may also be evaluated recursively, so that the integers  $A_i$  and  $B_i$  such that  $A_i/B_i = [b_0, b_1, \dots, b_i]$  may be calculated efficiently. For factorization, the relation

$$A_{i-1}^2 \equiv (-1)^i Q_i \pmod{N} \tag{3}$$

has been important for several algorithms. This equation also explains the name 'pseudo-square'.

Take the following example:

#### Example 4

$$\begin{array}{ll}
x_0 = \sqrt{41}, \quad b_0 = 6 & \sqrt{41} = 6 + \frac{1}{x_1} \\
x_1 = \frac{1}{x_0 - b_0} = \frac{1}{\sqrt{41} - 6} = \frac{\sqrt{41} + 6}{5}, \quad b_1 = 2 & \sqrt{41} = 6 + \frac{1}{2 + \frac{1}{x_2}} \\
x_2 = \frac{1}{x_1 - b_1} = \frac{5}{\sqrt{41} - 4} = \frac{\sqrt{41} + 4}{5}, \quad b_2 = 2 & \sqrt{41} = 6 + \frac{1}{2 + \frac{1}{2 + \frac{1}{x_3}}} \\
x_3 = \frac{1}{x_2 - b_2} = \frac{5}{\sqrt{41} - 6} = \frac{\sqrt{41} + 6}{1}, \quad b_3 = 12 & \sqrt{41} = 6 + \frac{1}{2 + \frac{1}{2 + \frac{1}{12 + \frac{1}{x_4}}}}
\end{array}$$

At this point it is evident that  $x_4 = x_1$  and the sequence is periodic. This always occurs with the continued fraction expansion for  $\sqrt{N}$  for  $N$  non-square. The length of this period is closely related to several deep number theory ideas involving the class number and the regulator.

2. A *binary quadratic form* is a polynomial of the form  $F(x, y) = ax^2 + bxy + cy^2$ , for  $a, b, c \in \mathbb{Z}$ . (Often this is abbreviated<sup>5</sup> as  $(a, b, c)$ ).

Two quadratic forms are *equivalent* if they have the same range, the set of possible values for  $F(x, y)$  for integer values of  $x$  and  $y$ , written  $F_1 \sim F_2$ . §4 of Appendix A gives a more precise description of equivalence. The surprising fact is that two quadratic forms are equivalent if and only if they correspond to terms from the same continued fraction expansion (Theorem 6 in Appendix A), so that the cycle formed by the  $x_i$ 's used to compute continued fractions corresponds to a cycle of equivalent quadratic forms.

The primary operation on quadratic forms is *composition*. This operation is defined in detail in Appendix A, Proposition 2. The composition of two binary quadratic forms is another binary quadratic form, written with the symbol  $(*)$ . Note that although the original definition of this operation is related to multiplication, this is an operation that is quite distinct from multiplication, as the product of two binary quadratic forms is no longer a binary quadratic form.

Equivalence and composition are closely related. Specifically, if  $F_1 \sim F_2$ , then  $F_1 * G \sim F_2 * G$ . The cycles of equivalent forms are the elements of the *class group*, which has been studied extensively.

3. A *lattice* is the set of all finite linear combinations (with integer coefficients) of a generating set that contains a basis for the entire space the lattice is contained in. For this research, the elements of a lattice are vectors in  $\mathbb{Q}(\sqrt{N}) \times \mathbb{Q}(\sqrt{N})$ . If  $\alpha_1, \alpha_2, \dots, \alpha_k \in \mathbb{Q}(\sqrt{N}) \times \mathbb{Q}(\sqrt{N})$  are vectors, then

---

<sup>5</sup>As  $b$  is usually even, many writings on quadratic forms represent this as the triplet  $(a, b/2, c)$ , so that  $-14x^2 + 10xy + 5y^2$  would be represented  $(-14, 5, 5)$ . This report does not take out this factor of 2. Although this will be clear throughout this report, please observe the potential for confusion on this “annoying little 2 [2]” when comparing different sources on quadratic forms.

$$[\alpha_1, \alpha_2, \dots, \alpha_k] = \left\{ \sum_{i=1}^k n_i \alpha_i : n_i \in \mathbb{Z} \right\}$$

is a lattice.

The notation is often abused slightly, so that the vector  $\langle \zeta, \bar{\zeta} \rangle$  is represented by just  $\zeta$ . There are several properties of lattices that are important for this research.

**Definition 1** For a vector  $v = \langle v_1, v_2 \rangle$ , the *normed body* of  $v$ ,  $\mathcal{R}(v)$  is the set

$$\mathcal{R}(v) = \{ \langle x_1, x_2 \rangle : x_1, x_2 \in \mathbb{R}, |x_1| < |v_1|, |x_2| < |v_2| \}$$

Abusing notation again, denote  $\mathcal{R}(\xi) = \mathcal{R}(\langle \xi, \bar{\xi} \rangle)$ .

A number  $\xi$  (or actually the corresponding vector) is a *minimum* of  $\mathcal{L}$  if  $\mathcal{R}(\xi) \cap \mathcal{L} = \{0\}$ , where 0 is the vector  $\langle 0, 0 \rangle$ . Figure 1 provides an example of this.

A lattice  $\mathcal{L}$  is *reduced* if  $1 \in \mathcal{L}$  and 1 is a minimum.

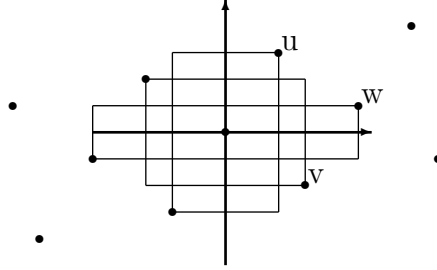


Figure 1: A Lattice with Minima

The image above demonstrates minima of a lattice. The lattice is the set of points. Each of the points  $u$ ,  $v$ , and  $w$  are minima of this lattice, since there are no points of the lattice inside their respective normed bodies (the rectangles) except the origin.

Another important characteristic of minima is adjacency. If  $\langle x_1, y_1 \rangle$  and  $\langle x_2, y_2 \rangle$  are minima with  $|x_1| > |x_2|$  and  $|y_1| < |y_2|$ , these two minima are *adjacent* if there does not exist another minima  $\langle x_3, y_3 \rangle$  such that  $|x_2| < |x_3| < |x_1|$  and  $|y_1| < |y_3| < |y_2|$ . In Figure 1,  $u$  and  $v$  are adjacent minima and  $v$  and  $w$  are adjacent minima.

The important connection between lattices and continued fractions is a method developed by Voronoi for finding a chain of adjacent minima for a lattice (and a chain of other reduced lattices). This process matches the continued fraction algorithm exactly, a property that implies that the distance derived from lattices also applies to continued fractions and quadratic forms.

4. An ideal is a common algebra concept that is used in a variety of contexts. For a ring  $S$ , a set  $I$  is an *ideal* of  $S$  if

$$I \subset S.$$

If  $a, b \in I$ , then  $a + b \in I$  and  $a - b \in I$ .

If  $a \in I$  and  $s \in S$ , then  $a \cdot s \in I$ .

A classic example is the set of all even integers,  $2\mathbb{Z}$ , within the set  $\mathbb{Z}$  of all integers. The sum of two even integers is still an even integer and the product of an even integer with any integer is an even integer.

This analysis focuses on ideals of  $\mathbb{Z}[\sqrt{N}]$ , for  $N$  a non-square positive integer. Describing these ideals will again require the notation for the lattice generated by a set. With this notation the ideals of  $\mathbb{Z}[\sqrt{N}]$  are of the form  $[Q, s\sqrt{N} + P]$ , for some integers  $Q, s, P$  and a set  $I = [Q, s\sqrt{N} + P]$  is an ideal of  $\mathbb{Z}[\sqrt{N}]$  if and only if  $sQ \mid \mathcal{N}(s\sqrt{N} + P)$ ,  $s \mid Q$ , and  $s \mid P$  (Appendix A, Lemma 8). An ideal is *primitive* if  $s = 1$ .

The multiplication of ideals is important to many fields. For the ideals  $I = [Q, \sqrt{N} + P]$  and  $J = [Q', \sqrt{N} + P']$ ,

$$I \cdot J = [QQ', Q\sqrt{N} + QP', Q'\sqrt{N} + Q'P, (\sqrt{N} + P)(\sqrt{N} + P)]$$

This may then be simplified to reduce it to an ideal of the form  $s[q, \sqrt{N} + p]$ , where  $s, q$ , and  $p$  may be calculated efficiently.

The quadratic form (of discriminant  $\Delta \equiv 0 \pmod{4}$ )  $F(x, y) = Ax^2 + Bxy + Cy^2$  corresponds to the ideal

$$\left[ A, \sqrt{\left(\frac{B}{2}\right)^2 - AC} + \frac{B}{2} \right]$$

and similarly the ideal  $[Q, \sqrt{N} + P]$  corresponds to the quadratic form

$$F(x, y) = Qx^2 + 2Pxy + \left(\frac{P^2 - N}{Q}\right)y^2$$

Note that  $\Delta = 4N$ .

The value of ideals is that the equations defining multiplication of ideals correspond exactly to the equation defining composition of quadratic forms, providing a connection between quadratic forms and lattices.

Appendix A describes the one to one correspondence between the elements of these different sets. For this paper, it is sufficient to understand that these maps do exist, that is, each term in the continued fraction expansion is paired with exactly one quadratic form, exactly one lattice, and exactly one ideal and given any one, any of these four expressions

may be calculated immediately. The indices derived from continued fractions are typically also used for the related quadratic forms, ideals, and lattices.

As all of these tools are closely related, they all have the property of being periodic, as demonstrated by Example 4. The length of this period is equal to the regulator of the class group and is related to many other number theory concepts ([15], [27]).

If the continued fraction expansion is started with something other than  $x_0 = \sqrt{N}$ , the result may be a sequence that is completely disjoint from the *principal* cycle (the sequence that begins with  $x_0 = \sqrt{N}$ ). Most of these cycles are symmetric<sup>6</sup>. Lemma 4 and Theorem 4 provide a thorough definition and analysis of these symmetries, but the concept can be readily understood from several examples:

**Example 5** *In Example 4, the period of the denominators was 1, 5, 5, 1, ..., so that 1 and 5 are both symmetry points. If  $x_0 = \sqrt{115}$ , then the period of the denominators is*

$$1, 15, 6, 11, 9, 10, 9, 11, 6, 15, 1, \dots$$

*so that the symmetry points are 1 and 10.*

*If instead  $x_0 = \frac{\sqrt{115}+9}{2}$ , then the period of the denominators is*

$$2, 17, 3, 5, 3, 17, 2, \dots$$

*This is a different cycle for the class group for  $N = 115$ .*

These are the cycles that are important for factorization.

The square of a quadratic form in the principal cycle is also in the principal cycle and Theorem 8 shows that the square of a form in any ambiguous cycle is also in the principle cycle.

### 3 History: From Morrison and Brillhart to Present

In 1931, D. H. Lehmer and R. E. Powers [13] introduced one simple and fairly intuitive algorithm for using continued fractions for factorization. Unfortunately, as it had the frustrating tendency to fail a few times before succeeding, it was dropped. However, as better computing capabilities became available, John Brillhart began to consider that although this algorithm was cumbersome for smaller numbers, it might have an advantage for larger numbers. In 1970, Morrison and Brillhart implemented CFRAC, as it became known, and tried to factor a 39 digit number [19]. The entire algorithm is a bit more complicated, but here is a description sufficient for our purposes.

If the equation

$$x^2 \equiv y^2 \pmod{N} \tag{4}$$

can be solved such that

---

<sup>6</sup>The non-symmetric cycles are not analyzed in this paper.

$$x \not\equiv \pm y \pmod{N}, \quad (5)$$

then  $\gcd(x - y, N)$  provides a nontrivial factor of  $N$ . Choosing a value for  $x$  and then looking for a value for  $y$  that satisfies equations (4) and (5) is not computationally effective for large  $N$ . Fermat, the first to employ this concept, tested values of  $x$  greater than  $\sqrt{N}$  to find a value such that  $x^2 \pmod{N}$  was already a perfect square. However, these numbers get large quickly. Continued fractions provide another approach to achieving this and since  $0 < Q_i < 2\sqrt{N}$ , the chances of finding a perfect square are greatly improved. From (3),  $A_{i-1}^2 \equiv (-1)^i Q_i \pmod{N}$ . If for some  $i$ ,  $Q_{2i}$  is a perfect square then this provides a solution to (4) and it only remains to check whether or not  $\gcd(x + y, N)$  or  $\gcd(x - y, N)$  provide a nontrivial factor of  $N$ . However, there are not very many perfect squares in the continued fraction expansion so Morrison and Brillhart [19] used products to obtain squares. For example, for  $N = 1333$ , the continued fraction expansion provides  $73^2 \equiv -3 \pmod{N}$  and  $1789^2 \equiv -12 \pmod{N}$ . From this,  $(73 \cdot 1789)^2 \equiv (-3)(-12) = 6^2 \pmod{N}$ , quickly yielding  $\gcd(73 \cdot 1789 - 6, 1333) = 43$ , so that  $1333 = 31 \cdot 43$ .

There are two problems with this algorithm. First, it requires the calculation of the  $A_i$ 's, which are of the same size as  $N$ , after reduction modulo  $N$ , while the rest of the algorithm only requires arithmetic on numbers of size  $\sqrt{N}$ . Second, after going through a nontrivial amount of computation to find a relation that solves (4), not all of these result in a factorization. I provide one example:

**Example 6** *In the continued fraction for  $\sqrt{1333}$ ,  $Q_6 = 9 = 3^2$  and  $A_5 = 10661$ , so that  $10661^2 \equiv 3^2 \pmod{1333}$ . Unfortunately  $10661 \equiv -3 \pmod{1333}$ , so this square does not result in a nontrivial factor of 1333.*

In 1975, Daniel Shanks developed several very interesting algorithms for factorization ([26],[29]) based on an understanding of quadratic forms and the class group infrastructure. First he developed an improvement to the Morrison-Brillhart algorithm. Roughly speaking, rather than saving the  $A_i$ 's, he was able to use composition of quadratic forms to combine numbers to produce squares and then use the “infrastructure” to use those squares to find a factorization. In addition, he developed from the concept of infrastructure a system of predicting whether or not any given square would provide a nontrivial factor. Unfortunately, this didn't save very much time and was a much more complicated algorithm than the Morrison-Brillhart algorithm.

From here the development of the algorithm was prompted by the number  $2^{60} + 2^{30} - 1$ . It failed a Fermat primality test<sup>7</sup>, but when Morrison and Brillhart tried to factor it, it failed 114 times. Therefore, they stopped, multiplied it by some small constant and tried again. This time it worked on the first try, but they wanted to know why it had failed so many times. So they asked Shanks to analyze it. Unfortunately (or fortunately in hindsight), Shanks only had an HP-65 available and he couldn't fit his entire algorithm into it. Therefore, he discarded all the work of combining numbers to form squares and just cycled through until

---

<sup>7</sup>The Fermat primality test is based on Fermat's classical result that for  $p$  prime,  $a^p \equiv a \pmod{p}$  [9]. Equivalently, if  $a^N \not\equiv a \pmod{N}$  for some integer  $a$ , then  $N$  isn't prime.

he found one already there. The code for this was much shorter, and as it turned out the algorithm, which became known as Square Forms Factorization or SQUFOF, was actually significantly faster.

Although Shanks did not publish his work, it has been picked up by several people. Williams [32], in 1985, provides a description of SQUFOF based on ideals and lattices. Riesel [23], in 1985, gives a good continued fraction based description of SQUFOF. Buell [2], in 1989, provides a thorough analysis of quadratic forms and gives a description of SQUFOF. No complete explanation of why SQUFOF works has been published.

Shanks died in 1996 leaving several papers on Square Forms Factorization in unpublished and incomplete form [33]. These were incomplete and contained very few proofs, although Shanks suggests that he may have had proofs for some of what he wrote. This author has obtained a copy of these papers, has transcribed them, and has formalized the proofs necessary to explain why Square Forms Factorization works. See Appendix A for the details of this analysis.

Recently, a runtime analysis of SQUFOF, among other things, has been completed by Jason Gower in [8], proving that the runtime is  $O(\sqrt[4]{N}) = L(1, 1/4)$ .

## 4 Square Forms Factorization

### Original Algorithm and Variations

The continued fraction cycles contain symmetry points that potentially provide a factorization for  $N$ . The goal of SQUFOF and any related algorithms is to search for these symmetry points. SQUFOF, described in Figure 2, provides a fairly simple way of achieving this. The basic version searches the sequence in order until it finds a perfect square on an even index. Then it takes the square root of the denominator and the conjugate of the numerator and continues cycling until it finds the symmetry point, which is indicated by a repeated numerator. At this point  $P_i$  at this point must then have a factor in common with the  $Q_{i+1}$ . Since  $N = P_i^2 + Q_i Q_{i+1}$ , this repeated numerator provides the factor for  $N$ .

**Example 7** *Let  $N$  be 1353:*

$$\begin{aligned} x_0 &= \sqrt{1353} \quad b_0 = 36 \\ x_1 &= \frac{1}{\sqrt{1353} - 36} = \frac{\sqrt{1353} + 36}{57} = 1 + \frac{\sqrt{1353} - 21}{57} \\ x_2 &= \frac{57}{\sqrt{1353} - 21} = \frac{\sqrt{1353} + 21}{16} = 3 + \frac{\sqrt{1353} - 27}{16} \end{aligned} \tag{6}$$

*The second fraction in each step is found by rationalizing. At each step, the integers taken out are  $b_i$  and the remaining fractions are between 0 and 1. After subtracting  $b_i$  the*

Given  $N$  to be factored:

$x_0 \leftarrow \sqrt{N}$   $b_0 \leftarrow \lfloor x_0 \rfloor$

**while**  $Q_i \neq \text{perfect square}$

Apply equation(2) twice

Reduce  $x_i$  to the form  $\frac{\sqrt{N}-P_i}{Q_i}$

$x'_0 \leftarrow \frac{\sqrt{N}+P_i}{\sqrt{Q_i}}$

**while**  $P_i \neq P_{i-1}$

Apply equation (2) and reduce

$\gcd(N, P_i)$ , the greatest common divisor, is a nontrivial factor.

Figure 2: SQUFOF without Composition

remaining fraction is inverted to find  $x_{i+1}$ . As a point of reference, we have approximated so far that  $\sqrt{1353} \approx 36 + \frac{1}{1+\frac{1}{3}}$ . SQUFOF stops here because 16 is a perfect square. Taking the conjugate of the numerator and the square root of the denominator, we obtain:

$$\begin{aligned}
 x'_0 &= \frac{\sqrt{1353} + 27}{4} = 15 + \frac{\sqrt{1353} - 33}{4} \\
 x'_1 &= \frac{4}{\sqrt{1353} - 33} = \frac{\sqrt{1353} + 33}{66} = 1 + \frac{\sqrt{1353} - 33}{66}
 \end{aligned} \tag{7}$$

Here, since the residue 33 is repeated, we quickly find that 33 is a factor of 1353.  $1353 = 33 \cdot 41 = 3 \cdot 11 \cdot 41$ .

Shanks developed Square Forms Factorization, or SQUFOF, based on a concept called the *infrastructure* of the class group. This refers to the relationship between distance and composition. Define the *distance* from  $F_m$  to  $F_n$  by:

$$D(F_m, F_n) = \log\left(\prod_{k=m+1}^n x_k\right) \tag{8}$$

where  $x_k$  are the corresponding terms in the continued fraction expansion. Since distance is the log of the product of the terms in between, roughly speaking, distance is nearly proportional to the difference in indices. Then the location of the form resulting from the composition of two quadratic forms is controlled by:

$$D(F_1 * G_1, F_n * G_m) = D(F_1, F_n) + D(G_1, G_n) + \zeta \tag{9}$$

where  $\zeta$  is small<sup>8</sup>. This formula is proven in the arguments leading up to Theorem 11.

By equation (9), the distance from a symmetry point is doubled when a quadratic form is squared, so that the square root of a quadratic form is half the distance from a symmetry point as the original quadratic form. The change made to the continued fraction expansion upon the occurrence of a perfect square corresponds exactly to taking the square root of a quadratic form and reversing its direction, so that in the second phase (after the change in sequence is made) the algorithm is going backwards on a different cycle. The distance covered from making this change until obtaining the symmetry point will be one half the distance covered before finding this square. Since this distance may be known as well as necessary, it is possible to use some of the quadratic forms found along the way to shorten this return.

Formally, here is the version of Square Forms Factorization that Shanks intended:

Given  $N$  to be factored:

$Q_0 \leftarrow 1, P_0 \leftarrow \lfloor \sqrt{N} \rfloor, Q_1 \leftarrow N - P_0^2$

$r \leftarrow \lfloor \sqrt{N} \rfloor$

**while**  $Q_i \neq$  perfect square for some  $i$  even

$b_i \leftarrow \left\lfloor \frac{r+P_{i-1}}{Q_i} \right\rfloor$

$P_i \leftarrow b_i Q_i - P_{i-1}$

$Q_{i+1} \leftarrow Q_{i-1} + b_i(P_{i-1} - P_i)$

**if**  $i = 2^n$  for some  $n$

Store  $(Q_i, 2 \cdot P_i)$   $F_0 = (\sqrt{Q_i}, 2 \cdot P_{i-1}, \frac{P_{i-1}^2 - N}{Q_i})$

Compose  $F_0$  with stored forms according to the binary representation of  $i/2$  and store result to  $F_0$ .

$F_0 = (A, B, C)$

$Q_0 \leftarrow |A|, P_0 \leftarrow B/2, Q_1 \leftarrow |C|$

$q_0 \leftarrow Q_1, p_0 \leftarrow P_0, q_1 \leftarrow Q_0$

**while**  $P_i \neq P_{i-1}$  and  $p_i \neq p_{i-1}$

Apply same recursive formulas to  $(Q_0, P_0, Q_1)$  and  $(q_0, p_0, q_1)$

If  $P_i = P_{i-1}$ , either  $Q_i$  or  $Q_i/2$  is a nontrivial factor of  $N$ .

If  $p_i = p_{i-1}$ , either  $q_i$  or  $q_i/2$  is a nontrivial factor of  $N$ .

Figure 3: SQUFOF with Composition

Note that it is faster to approximate roughly which quadratic forms are needed to find the symmetry point than to actually calculate the distance exactly. The result is that after this composition, the quadratic form is close to the symmetry point but not exactly on it. In the last **while** loop, the search for the symmetry point has to be made in both directions,

<sup>8</sup>The bound for  $\zeta$  is proportional to  $\log \log N$  while the two distances on the right side of the formula have a bound proportional to  $\log N$ , so  $\zeta$  is usually negligible, although it may be calculated if necessary.

but this search is typically very short. The only forms that need to be stored are those with an index that is a power of two.

For example, if a square were found on step 56, there should be roughly 28 steps backward in the second phase. If after taking the square root and reversing the direction,  $F_0$  is composed with the quadratic form with index 16, the form with index 8, and the form with index 4, the result will be very close to the symmetry point.

Although much of the preceding theory was known, some of it was not and most of it was very scattered with few proofs. In Appendix A I have compiled all of this information and provided proofs and examples. Lemmas 1 - 2 provide a new and intuitive explanation of the reversal of direction effected by taking the conjugate of the numerator. Lemmas 3 - 4 use this understanding to analyze the periodicity and symmetries of the cycles. Lemma 5 formally proves a well-known intuitive result about reduction. Lemma 6 analyzes the symmetry points of the quadratic form cycle. Theorem 8 provides an important result on how ambiguous cycles fit in the class group. This result was probably understood by Shanks, as it is important for SQUFOF, but apparently never stated explicitly. Lemma 8 provides some slight generalization to the description of ideals. Typically  $s = 1$ . This slight extra generality allows a proof that multiplication of ideals corresponds to composition of quadratic forms. Theorem 9 provides a slight generalization of a well known result. Typically,  $h = 1$ , but the inclusion of the possibility that  $h = 2$  allows the results from the distance formula to be generalized to quadratic forms of discriminant  $\equiv 1 \pmod{4}$  without working in  $\mathbb{Z} \left[ \frac{\sqrt{N}+1}{2} \right]$ . Lemmas 11 and 12 are derived from [32] but have some variations that leads more directly to the method of Voronoi and make it easier and more intuitive to prove that lattices correspond to continued fractions. Lemma 14 uses the connection with continued fractions to derive a distance for reduction.

Theorem 11 relates composition of quadratic forms with distance. Williams [32] provides a very special case of this (limited to the principal cycle). This generalization is essential for SQUFOF as it relates distances in different cycles with composition. Lemma 15, which relates the entire distance around different cycles, is proved based on this.

§8 provides a complete description of Square Forms Factorization, including an explanation of why Shanks was able to know whether or not a square was proper and an explanation of what Shanks referred to as Fast Return. Fast Return, an idea that is typically not implemented with SQUFOF, uses composition to significantly speed up the second phase of the algorithm.

Appendix B includes the Magma implementation of SQUFOF with Fast Return. These algorithms have also been implemented in C. These have both been tested extensively for functionality. The runtimes have also been compared to elliptic curve method, multiple polynomial quadratic sieve, and the number field sieve. Although a thorough comparison of SQUFOF with other algorithms has not been done, it is fairly clear that on a single computer SQUFOF is slower than elliptic curves, the number field sieve, or the multiple polynomial quadratic sieve for most numbers. However, it has been conjectured by Pomerance [21] that a parallel implementation of SQUFOF would be competitive. This is the current direction of research.

## Parallelization

The current trend in ultra-fast computing is to use a large number of inexpensive processors instead of a single expensive processor [11]. With the large amount of computation required for factorization, the efficiency of a parallel implementation is especially important for these algorithms. In 1982, Shanks and Cohen attempted a parallelization by having multiple processors attempt to factor  $N$ ,  $3N$ ,  $5N$ , etc. This author has not attempted this parallelization yet, but reportedly it was not effective. Gower's recent Ph.D. thesis (under Dr. Wagstaff) [8] analyzed the use of multipliers and found them effective, but not necessarily as a means of parallelization (i.e. It is beneficial on average to multiply  $N$  by several small factors before factoring, but working on different multiples of  $N$  simultaneously does not provide a significant advantage.)

Composition of quadratic forms provide a different possible approach to parallelization. By computing a quadratic form several steps into the cycle and squaring it a reasonable<sup>9</sup> number of times, a quadratic form far out into the cycle can be found. Call it  $F$ . The first processor may be assigned to search this segment for a perfect square. The next processor can search from  $F$  to  $F^2$ , the next from  $F^2$  to  $F^3$ , etc. Each of these segments will be roughly equal sizes. When a processor finds a perfect square, it may use the quadratic forms involved in computing its segment in order to come close to the symmetry point. Specifically, this last step is made easier if the starting point of the segment is found by squaring. In pseudo-code:

Given  $N$  to be factored:

$r \leftarrow \lfloor \sqrt{N} \rfloor$

$F_0 \leftarrow (1, 2r, N - r^2)$

Cycle  $F_0$  several steps forward.

**for**  $i = 1$  to size (size is the logarithmic size of a segment.)

$F_i \leftarrow F_{i-1} * F_{i-1}$

Processor 0:

Assign one processor to search from  $F_0$  to  $F_{size}$ .

$F_{start} \leftarrow F_{size}$

$F_{end} \leftarrow F_{size}^2$

$F_{rootS} \leftarrow F_{size-1}$

$F_{rootE} \leftarrow F_{size}$

$F_{step} \leftarrow F_{size-1}$

**while** A factor hasn't been found

Wait for a processor to be free and send  $F_{start}$ ,  $F_{end}$ , and  $F_{rootS}$ .

$F_{start} \leftarrow F_{end}$

$F_{rootS} \leftarrow F_{rootE}$

$F_{rootE} \leftarrow F_{rootE} * F_{step}$

$F_{end} \leftarrow F_{rootE}^2$

---

<sup>9</sup>Empirically, 20-30 times works nicely. This parameter isn't critical.

Processor  $n$ :

Recieve  $F_{start}$ ,  $F_{end}$ , and  $F_{rootS}$

count  $\leftarrow 0$

$F_0 = (A, B, C)$

**while** A factor is not found and  $F_{start} \neq F_{end}$

    Cycle  $F_{start}$  forward 2 steps.

    count  $\leftarrow$  count+1

**if** A is a perfect square

$F_{test} \leftarrow F_{start}^{-1/2}$

$F_{test} \leftarrow F_{test} * F_{rootS}$

**for**  $j = size$  to 1

**if** count  $> 2^j$

$F_{test} \leftarrow F_{test} * F_j$

                count  $\leftarrow$  count  $- 2^j$

        Search in both directions from  $F_{test}$  for a symmetry point.

**if** Factorization found at symmetry point, output and quit.

**if** A factor is still not found

    Recieve new  $F_{start}$ ,  $F_{end}$ , and  $F_{rootS}$  and start over.

(This loop composes  $F_{test}$  with the necessary forms to bring it close to the symmetry point.)

Figure 4: Parallel SQUFOF

Since there is no overlap between the segments searched by the processors and since the perfect squares are, apparently, distributed randomly, this parallelization should be extremely efficient. The size of the segments is determined by how far the first **for** loop goes. There are only two hazards when choosing this. If the segment size is too small, the processors will finish their segments so quickly that recieving new segments will become a bottleneck. Alternately, if the segments are too long, the processors may divide up more than the entire cycle, so that there is overlap. However, except for rare numbers that will factor trivially fast regardless, there is significant room in between these two bounds.

This parallel version has been implemented in C using the MPI library. The code is included in Appendix C. It has not yet been compared with other algorithms.

## Factorization using a Test of Direction

Based on this symmetry of the cycles of quadratic forms, any fast test to determine whether or not the continued fraction expansion is in the correct direction, that is, whether or not it has not passed the factorization yet, would provide a faster factorization algorithm by performing a binary search (Figure 5).

```

Given  $N$  to be factored:
 $Q_0 \leftarrow 1, P_0 \leftarrow \lfloor \sqrt{N} \rfloor$ 
Apply equation (2) for 2 steps.
 $F_0 \leftarrow (Q_2, 2 \cdot P_2, -Q_3)$ 
 $i \leftarrow 0$ 
while  $F_i$  is in the right direction
     $F_{i+1} \leftarrow F_i * F_i$ 
     $i \leftarrow i + 1$ 
 $F_{last} \leftarrow F_{i-1}$ 
while  $i \geq 0$ 
    if  $F_{last} * F_i$  is in the right direction
         $F_{last} \leftarrow F_{last} * F_i$ 
     $i \leftarrow i - 1$ 
Search forward from  $F_{last}$  (should be only a couple steps)
to obtain the factorization.

```

Figure 5: Binary Search based on a Test of Direction

**Example 8**  $N = 2035153$ ,  $Q_0 = 1$ ,  $P_0 = 1426$

Then,  $F_0 = (1, 2 \cdot 1426, -1677)$ . The square of this form would be itself, so we recursively step forward a few steps to obtain:  $F_8 = (663, 2774, -168)$ .

This form is squared until it is past the symmetry point:

$$\begin{aligned}
 F_8 * F_8 &= (1569, 1522, -928) = F_{22} \\
 F_{22} * F_{22} &= (896, 2798, -87) = F_{44} \\
 F_{44} * F_{44} &= (1648, 1726, -783) = F_{82} \\
 F_{82} * F_{82} &= (411, 2104, -2259) \text{ is past the symmetry point.} \\
 F_{82} * F_{44} &= (9, 2846, -1136) = F_{134} \\
 F_{134} * F_{22} &\text{ is past the symmetry point.} \\
 F_{134} * F_8 &= (1153, 1898, -984) = F_{144}
 \end{aligned}$$

Recursively stepping forward from here, it is quickly found that  $F_{147} = (-1008, 2018, 1009)$  and  $F_{148} = (1009, 2018, -1008)$ . Therefore, 1009 is the symmetry point and  $2035153 = 1009 \cdot 2017$ .

The decisions for this example of whether or not a form was reversed (past the symmetry point) were determined by merely comparing with the actual continued fraction expansion. However, ideally this can be done without doing the entire expansion. The function that determines whether or not a form is reversed is called a *test of direction*. There are several possible candidates:

**Conjecture 1** If  $Q_i | Q_{i-1}$ ,  $(Q_i)^3 \nmid Q_{i-1}$ , and  $Q_i$  is not a power of 2, then the continued fraction expansion is in the correct direction.

In all other cases, this test provides no information. The condition  $Q_i | Q_{i-1}$  does not occur very frequently, so on its own it cannot provide a useful test of direction. However, the attempt to explain this empirical pattern has provided several possible alternatives:

First, it is possible to find a multiple  $k$  such that in the expansion with  $Q_i$  unchanged,  $P_i$  replaced by  $kP_i$  and  $N$  replaced by  $k^2N$ , this condition is met. For notation, let  $Q_i = Q'_0$ . Since using  $-k$  instead of  $k$  produces the same sequence in the opposite direction, it is necessary to relate this sequence to the original sequence in order to extract useful information. From this, it has so far been found that if  $Q_{i+3} | Q'_1$  and  $Q'_0 | Q'_{-1}$ , then the original expansion is in the correct direction. Finding multiples such that the second condition is satisfied can be accomplished by using continued fractions to find the convergents of  $\frac{\sqrt{N}-R_i}{Q_i^2}$ , where  $R_i \equiv \sqrt{N} \equiv P_i - \frac{P_i^2-N}{2P_i} \pmod{Q_i^2}$ .

This provides a test of direction that returns a result about 2% of the time. This would provide a polynomial time factorization algorithm, except that the time required to do this test of direction increases with the size of  $N$ .

One other possible test of direction involves *reduction distance*. After a quadratic form is squared, it often requires multiple steps in order to reduce to a quadratic form within the standard bounds. This is referred to as reduction distance. Since the overall distance from a symmetry point is doubled when a quadratic form is squared, it makes some intuitive sense that this distance from the nearest symmetry point should be related to this reduction distance. Empirically, the reduction distance for a quadratic form oriented away from the nearest symmetry point tends to be shorter. Concerning the conjecture, if  $Q_i | Q_{i-1}$ , then the square of  $(Q_i, P_{i-1}, -Q_{i-1})$  is  $(Q_i^2, P_{i-1}, -Q_{i-1}/Q_i)$  and the reduction distance is 0, so this would provide an explanation for Conjecture 1.

Empirically, comparing reduction distances is a test of direction that generally works near the symmetry point, but is inconclusive away from the symmetry point. Thus, one other idea may be necessary in order to apply this. Using the *class number formula* [5], it is possible to approximate what the distance to the symmetry point should be. This approximation may be able to come close enough to the symmetry point that some combination of the two tests of direction may be able to provide the necessary information.

## Secure and Insecure Numbers

It is possible for the entire cycle to not contain a single perfect square that provides a factorization, but this is extremely rare. There are some cases where it is possible to prove that the algorithm will work.

**Proposition 1** *Let  $N \equiv 1 \pmod{4}$ , with  $-1$  not a quadratic residue of  $N$ . Let  $Q_s$  be the second symmetry point in the principal cycle.  $\gcd(Q_s, N)$  is a nontrivial factor of  $N$ .*

**Proof:** This proof is based on a number of results included in Appendix A.

By Theorem 4, there exists a symmetry point  $Q_s$  distinct from 1 and since  $-1$  is not a quadratic residue of  $N$ , the period must be even, so that  $Q_s | 2N$ . Assume  $Q_s = 2$ . By

Theorem 2 (a),  $N = P_s^2 + Q_s Q_{s+1}$ , so that  $P_s$  must be odd. Therefore, since  $N \equiv 1 \pmod{4}$ ,  $4 \mid (N - P_s^2)$ , so that  $2 \mid Q_{s+1}$ . Therefore, by induction using Theorem 2 (b) and (g),  $Q_i$  must be even and  $P_i$  must be odd for all  $i$ . However, this is the principal cycle, so  $Q_0 = 1$ , contradicting this. Therefore  $Q_s \neq 2$ . The only remaining possibility is that  $Q_s > 2$  and therefore, since  $Q_s \mid 2N$ ,  $\gcd(Q_s, N)$  is a nontrivial factor for  $N$ . **QED**

For most other numbers, there appears to be no classification or order to the few numbers that SQUFOF does not work for. However, it is also important to analyze which numbers SQUFOF is faster or slower on.

In addition to being poorly understood and incompletely implemented, Square Forms Factorization dropped from the forefront of research because it was conjectured that it was only fast on integers with a high class number. At first, this appears reasonable, as a high class number implies shorter period length and more squares. Indeed, integers with high class numbers do factor very quickly with SQUFOF. However, this is not a thorough description. For example  $100000000000037 \cdot 490000000000013$  has class number 2 and SQUFOF finds a factor on the second step. Conversely,  $100000000000037 \cdot 500000000000057$  also has class number 2 but does not contain a perfect square until after 2098040 steps, even though these two numbers are roughly the same size and have roughly the same size factors.

Although these numbers were specifically calculated to meet this criteria (the 0's do make them look a bit unique), the effect is not unique and there are numbers of every size that do this. An integer such as the first could have been chosen for the RSA algorithm without any indication that there was a problem.

The reason the first number factored quickly was that the ratio of the factors is close to a perfect square. Specifically, the second factor is approximately 49 times the first factor. In the second number, the second factor is approximately 50 times the first number. In general numbers of the form  $(a^2m + b)(c^2m + d)$  factor very quickly. However, it is not possible to put an exact bound on how large  $b$  and  $d$  can be in this form, as for any range that might be set for  $b$  and  $d$ , there are numbers that factor unusually fast and numbers that don't. Empirically, it appears that if  $b, d < \sqrt{m}$ , there is a high probability of this number factoring quickly. Conjecture 2 provides the conditions that appear sufficient and necessary for an integer to factor unusually fast.

**Conjecture 2** *Let  $N = pq$ . Let  $\frac{A_n}{B_n}$  be the convergents to  $\frac{p}{q}$  obtained from continued fractions. If for some  $n$ , both  $A_n$  and  $B_n$  are perfect squares, then the runtime for the factorization of  $N$  is  $O(\sqrt{\min(A_n, B_n)})$ .*

A proof, or even a really good explanation, for Conjecture 2 is still lacking. It is derived from the fact that Square Forms Factorization (and other algorithms related to continued fractions) may be considered as a generalization of Fermat's method (see §3). Fermat's method is fast for numbers whose factors are close to each other. Continued fractions generalize this to be fast whenever the factors may be multiplied by perfect squares to be close to each other. The condition that is set is a measure of how close the two numbers (after multiplication by squares) are to each other. This conjecture has been tested empirically and

found to be true on over 100000 integers ranging from 30 digits to over 100 digits. Average runtime appears to be less than a second for numbers that meet this criteria.

In the current implementations of the RSA algorithm, there are conditions on the primes chosen and the randomness of the process of choosing primes, but the threat that the two primes chosen may be related to each other in some way is not checked for [17]. It is the recommendation of this author that the RSA algorithm implement a quick check that the two primes do not meet the conditions of Conjecture 2.

# Bibliography

- [1] Alford, W. R., A. Granville, and C. Pomerance. “There are Infinitely Many Carmichael Numbers.” *Annals of Math.* v. 139. 1994. p. 703-722.
- [2] Buell, D. A. *Binary Quadratic Forms: Classical Theory and Modern Computations*. New York: Springer-Verlag, 1989.
- [3] Crandall, Richard and Carl Pomerance. *Prime Numbers: a Computational Perspective*. New York: Springer. 2001
- [4] Delone, B. N. and D. K. Faddeev. *The Theory of Irrationalities of the Third Degree*. American Mathematical Society. Providence, RI. 1964.
- [5] Davenport, Harold. *Multiplicative Number Theory*. Springer-Verlag. New York. 1980.
- [6] Ellis, J. H. “The history of non-secret encryption”. *Cryptologia*. Number 23, July 1999, p 267-273.
- [7] Gauss, Carl Friedrich. *Disquisitiones Arithmeticae*. Trans. by Arthur A. Clarke. New Haven, Yale University Press, 1966.
- [8] Gower, J. E. *Square Form Factorization*. Thesis. Purdue University. December 2004. <<http://www.cerias.purdue.edu/homes/ssw/gowerthesis804/wthe.pdf>>
- [9] Hardy, G. H. and Wright, E. M. *An Introduction to the Theory of Numbers*. Oxford: Clarendon Press. 1979.
- [10] Hua, L. K. *Introduction to Number Theory*. Springer-Verlag. New York. 1982.
- [11] Kuck, David J. et all. “Parallel Supercomputing Today and the Cedar Approach.” *Science*, New Series, V. 231, No. 4741, 1986, p. 967-974.
- [12] Lang, Serge. *Algebra*. Reading, Mass., Addison-Wesley Pub. Co. 1965.
- [13] Lehmer, D. H. and R. E. Powers. “On factoring large numbers.” *Bull. Amer. Math. Soc.* v. 37, 1931, p. 770-776.
- [14] Lenstra, A. K. et all “The development of the number field sieve.” *Lecture Notes in Mathematics*, 1554. Springer-Verlag, Berlin, 1993.

- [15] Lenstra, H. W. Jr. "On the Calculation of Regulators and Class Numbers of Quadratic Fields." London Mathematical Society. v. 56, 1980, p/ 123-150.
- [16] Lenstra, H. W. Jr. "Factoring integers with elliptic curves". *Annals of Mathemtaics*, 126. 1987. p 649-673.
- [17] Menezes, Alfred J. , Paul C. van Oorschot and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, October 1996.
- [18] Mollin, R. A. and A. J. van der Poorten. "A Note on Symmetry and Ambiguity". *Bull. Austral. Math. Soc.* Austral. Math. Publ. Assoc. 1995
- [19] Morrison, Michael A. and John Brillhart "A Method of Factoring and the Factorization of  $F_7$ ." *Mathematics of Computation*, Vol. 29, No. 129 (Jan, 1975), 183-205.
- [20] Pomerance, C. "A Tale of Two Sieves." *Notices American Mathematical Society* 43, 1473-1485, 1996.
- [21] Pomerance, C. Personal communcation to David Joyner. June 2005.
- [22] Poorten, Alfred J. "A Note on NUCOMP". *Mathematics of Computation*. Volume 72, Number 244, April 2003, p 1935-1946.
- [23] Riesel, Hans. *Prime Numbers and Computer Methods for Factorization*. Boston : Birkhuser, 1985. p 191-195, 300-317.
- [24] Rivest, R., A. Shamir and L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". *Communications of the ACM*, 21 (2), p 120-126, February 1978
- [25] Sartorius von Waltershausen, W. *Gauss: A Memorial*. Colorado Springs, Colorado. 1966.
- [26] Shanks, Daniel. "Analysis and Improvement of the Continued Fraction Method of Factorization." Unpub. circa 1975. Transcribed by Stephen McMath March 2004. <<http://cadigweb.ew.usna.edu/~wdj/mcmath/>>
- [27] Shanks, Daniel. "The Infrastructure of a Real Quadratic Field and its Applications." *Proceedings of the 1972 Number Theory Conference : University of Colorado, Boulder, Colorado, August 14-18, 1972.*
- [28] Shanks, Daniel. "On Gauss and Composition II". Ed. R. A. Mollin. *Number Theory and Applications*. Kluwer Academic Publishers. 1989. p 179-204.
- [29] Shanks, Daniel. "SQUFOF Notes." circa 1975. Unpub. Transcribed by Stephen McMath March 2004. <<http://cadigweb.ew.usna.edu/~wdj/mcmath/>>

- [30] Silverman, R. D. "The Multiple Polynomial Quadratic Sieve." *Math. Comput.* 48, 329-339, 1987.
- [31] Trappe, Wade and Lawrence C. Washington. *Introduction to Cryptography with Coding Theory*. New Jersey: Prentice-Hall, Inc. 2002. p 137-142.
- [32] Williams, Hugh C. "Continued Fractions and Number-Theoretic Computations." *Rocky Mountain Journal of Mathematics*. Volume 15, no. 2, Spring 1985.
- [33] Williams, H. C. "Daniel Shanks (1917-1996)". *Mathematics of Computation*. Volume 66, Number 219, July 1997, p 929-934.

# Index

- class group, 13
- composition of forms, 13
- congruent, 10
- conjugate, 11
- continued fractions, 12
- convergents, 12, 26
- correspondence, 15
- cryptography, 7
- cycles, 16
  
- distance, 14, 19
- divides ( $\mid$  or  $\parallel$ ), 10
  
- ECM, 9
- elements ( $\in$ ), 9
- Euclid's Algorithm, 10
  
- Fermat's little theorem, 11
- floor, 12
  
- greatest common divisor, 10
  
- ideal, 15
- infrastructure, 19
- integers ( $\mathbb{Z}$ ), 10
  
- lattice, 13
- least common multiple, 10
  
- modulo, 10
- MPQS, 9
  
- natural numbers, 10
- NFS, 9
- non-square, 12
- norm, 11
- normed body ( $\mathcal{R}$ ), 14
  
- parallelization, 9, 22
- perfect square, 17
- prime, 10
- primitive, 15
- pseudo-square, 12
  
- quadratic forms, 13
- Quadratic Reciprocity Law, 11
- quadratic residue, 7, 10
  
- rational numbers ( $\mathbb{Q}$ ), 11
- real numbers ( $\mathbb{R}$ ), 9
- relatively prime, 10
- residue, 12
- RSA, 7
- runtime, 8
  
- subset ( $\subset$ ), 9
- symmetry points, 16
  
- test of direction, 24

# Appendix A

## Daniel Shanks' Square Forms Factorization

This is a detailed expository account of Shanks' Square Forms Factorization (SQUFOF) method, including proofs or detailed references of all results, in light of an understanding of continued fractions, binary quadratic forms, lattices, and ideals.

### 1 Introduction

The problem of distinguishing prime numbers from composite numbers and of resolving the latter into their prime factors is known to be one of the most important and useful in arithmetic. It has engaged the industry and wisdom of ancient and modern geometers ... the dignity of the science itself seems to require that every possible means be explored for the solution of a problem so elegant and so celebrated.

C. F. Gauss [7]

There is a significant body of knowledge concerning quadratic forms, continued fractions, lattices, and ideals. However, much of this information is very spread out, especially that dealing with the class group “infrastructure”. Therefore, one major purpose here is to organize this information into a usable form, along with providing some of the connecting historical information and providing sufficient examples so that the average reader may be able to understand the main concepts, if perhaps not all of the minutia of the proofs.

These are extremely rich fields, and there are problems and ideas that have yet to be addressed concerning binary quadratic forms and even concerning variations of SQUFOF that Shanks considered. This paper is a complete proof of the simplest case only.

Of these objects, continued fractions, described in §2, appear the most concrete and are the easiest to examine examples of, especially with respect to distance, and thus are useful

for a conceptual understanding. §3 describes some of the direct applications of continued fractions and the problems that caused Shanks to develop the theory further. Quadratic forms, described in §4, are computationally the simplest format to implement and have given rise to composition, an extremely useful tool. Ideals, in §5 are valuable because they provide an alternate interpretation of composition and provide a link to lattices, described in §6, from which “distance” is derived. §7 uses lattices and ideals to prove Theorem 11, a powerful formula concerning infrastructure distance. §8 puts it all together to analyze Shanks’ factorization algorithm.

This investigation will also define mappings between these different objects. These maps will be represented by the letter  $\Phi$ , with subscripts indicating the two sets being considered. For example,  $\Phi_{\mathbb{T},\mathbb{F}}$  would be a map from terms in the continued fraction to quadratic forms and  $\Phi_{\mathbb{F},\mathbb{T}} = \Phi_{\mathbb{T},\mathbb{F}}^{-1}$  would be its inverse. Note that the order of the subscripts matters. Each of these maps will be addressed individually.

## 2 Continued Fractions

One tool used by many different algorithms is the continued fraction expression for  $\sqrt{N}$ , where  $N$  is the number to be factored. This expression is calculated recursively[9]:

$$x_0 = \sqrt{N}, \quad b_0 = \lfloor x_0 \rfloor \quad (\text{A.1})$$

$$\forall i \geq 1 \quad x_i = \frac{1}{x_{i-1} - b_{i-1}}, \quad b_i = \lfloor x_i \rfloor \quad (\text{A.2})$$

$$\sqrt{N} = b_0 + \frac{1}{b_1 + \frac{1}{b_2 + \dots}} \quad (\text{A.3})$$

Observe that solving equation (A.2) for  $x_{i-1}$  gives  $x_{i-1} = b_{i-1} + \frac{1}{x_i}$ . Repeatedly substituting this into itself gives equation (A.3).

Before developing the theory too much further, allow me to offer one example of how this works, just so that the pieces make sense to you. To simplify the expansion some, the integers taken out in the third step of each line are the  $b_i$ :

### Example 9

$$\begin{array}{ll} x_0 = \sqrt{41}, \quad b_0 = 6 & \sqrt{41} = 6 + \frac{1}{x_1} \\ x_1 = \frac{1}{\sqrt{41}-6} = \frac{\sqrt{41}+6}{5} = 2 + \frac{\sqrt{41}-4}{5} & \sqrt{41} = 6 + \frac{1}{2+\frac{1}{x_2}} \\ x_2 = \frac{5}{\sqrt{41}-4} = \frac{\sqrt{41}+4}{5} = 2 + \frac{\sqrt{41}-6}{5} & \sqrt{41} = 6 + \frac{1}{2+\frac{1}{2+\frac{1}{x_3}}} \\ x_3 = \frac{5}{\sqrt{41}-6} = \frac{\sqrt{41}+6}{1} = 12 + \frac{\sqrt{41}-6}{1} & \sqrt{41} = 6 + \frac{1}{2+\frac{1}{2+\frac{1}{12+\frac{1}{x_4}}}} \end{array}$$

From this step, it is evident that  $x_4 = x_1$  and the cycle repeats from here. Observe that if the sequence starts with  $x_0 = \sqrt{41} + 6$ , then also  $x_3 = x_0$ . Regardless, the expression on the right, if truncated at any point provides a rational approximation to  $\sqrt{41}$ . Often this will be written as merely  $[6, 2, 2, 12, \dots]$  to save space or  $[6, \overline{2, 2, 12}]$  to indicate that this part repeats. The various numbers on the left have some important properties that we will now analyze in some depth.

Throughout, assume that  $N$  is an odd positive integer and is not a perfect square. For number theory purposes, let

$$x_0 = \frac{\sqrt{N} + P_{-1}}{Q_0}$$

where  $P_{-1}, Q_0$  are integers chosen such that

$$P_{-1}^2 \equiv N \pmod{Q_0}, \quad 0 < P_{-1} < \sqrt{N}, \quad \text{and} \quad |\sqrt{N} - Q_0| < P_{-1}. \quad (\text{A.4})$$

There are many ways of doing this<sup>1</sup>. The recursive formulas are:

$$x_{i+1} = \frac{1}{x_i - b_i} \quad b_i = \lfloor x_i \rfloor, \quad i \geq 0$$

Formally, the assumed equation is:

$$x_{i+1} = \frac{Q_i}{\sqrt{N} - P_i} = \frac{\sqrt{N} + P_i}{Q_{i+1}} = b_{i+1} + \frac{\sqrt{N} - P_{i+1}}{Q_{i+1}}, \quad i \geq 0 \quad (\text{A.5})$$

Note that this equation serves as a definition of  $Q_i, P_i, Q_{i+1}, P_{i+1} \in \mathbb{Q}$ , so that these equations are true regardless of the conditions on these variables. Theorem 2 provides some well-known fundamental properties and identities of continued fractions. In [23], Hans Riesel provides very clear proofs of most of this.

**Theorem 2** [23] *In the continued fraction expansion of  $x_0$  satisfying (A.4), each  $x_i$  reduces to the form  $\frac{\sqrt{N} + P_{i-1}}{Q_i}$ , with (a)  $N = P_i^2 + Q_i Q_{i+1}$ , (b)  $P_i = b_i Q_i - P_{i-1}$ , (c)  $b_i = \left\lfloor \frac{\lfloor \sqrt{N} \rfloor + P_{i-1}}{Q_i} \right\rfloor \geq 1$ , (d)  $0 < P_i < \sqrt{N}$ , (e)  $|\sqrt{N} - Q_i| < P_{i-1}$ , (f)  $Q_i$  is an integer, and (g)  $Q_{i+1} = Q_{i-1} + b_i(P_{i-1} - P_i)$ . Furthermore, (h) this sequence is eventually periodic.*

**Proof:**

(a) From (A.5), the equation  $\frac{Q_i}{\sqrt{N} - P_i} = \frac{\sqrt{N} + P_i}{Q_{i+1}}$  requires that  $N = P_i^2 + Q_i Q_{i+1}$ .

(b) It is evident from simplifying the expression on the far right of (A.5) that

$$\frac{\sqrt{N} + P_i}{Q_{i+1}} = \frac{\sqrt{N} + b_{i+1} Q_{i+1} - P_{i+1}}{Q_{i+1}}.$$

---

<sup>1</sup>Choosing  $x_0 = \sqrt{N} + \lfloor \sqrt{N} \rfloor$ , so that  $P_{-1} = \lfloor \sqrt{N} \rfloor$  and  $Q_0 = 1$  is one possibility. Choosing  $x_0 = \frac{\sqrt{N} + P_{-1}}{2}$ , where  $P_{-1} = \lfloor \sqrt{N} \rfloor$  or  $\lfloor \sqrt{N} \rfloor - 1$ , such that  $P_{-1}$  is odd, is another possibility.

Therefore,  $P_{i+1} = b_{i+1}Q_{i+1} - P_i$ .

(c) For  $i = 0$ , by the assumption  $|\sqrt{N} - Q_0| < P_{-1}$

$$Q_0 < \sqrt{N} + P_{-1}$$

Therefore,

$$b_0 = \left\lfloor \frac{\sqrt{N} + P_{-1}}{Q_0} \right\rfloor \geq 1$$

For  $i > 0$ ,  $b_{i-1} = \lfloor x_{i-1} \rfloor$ . By the definition of floor,  $x_{i-1} - 1 < b_{i-1} \leq x_{i-1}$ . If  $b_{i-1} = x_{i-1}$ , then the continued fraction  $[b_0, b_1, \dots, b_{i-1}]$  is rational and is equal to  $x_0 = \frac{\sqrt{N} + P_{-1}}{Q_0}$ , which is irrational since  $N$  is not a perfect square. Therefore,  $x_{i-1} - 1 < b_{i-1} < x_{i-1}$ , so that  $0 < x_{i-1} - b_{i-1} < 1$ . Therefore,  $x_i = \frac{1}{x_{i-1} - b_{i-1}} > 1$ , so that  $b_i = \lfloor x_i \rfloor \geq 1$ .

Note that there is no integer between  $\lfloor \sqrt{N} \rfloor + P_{i-1}$  and  $\sqrt{N} + P_{i-1}$ , so it is trivial that  $\left\lfloor \frac{\sqrt{N} + P_{i-1}}{Q_i} \right\rfloor = \left\lfloor \frac{\lfloor \sqrt{N} \rfloor + P_{i-1}}{Q_i} \right\rfloor$ .

(d-e) The statements  $|\sqrt{N} - Q_i| < P_{i-1}$  and  $0 < P_{i-1} < \sqrt{N}$  may be proven inductively.

Base case:  $i = 1$

$P_0 = \lfloor \sqrt{N} \rfloor$  or  $\lfloor \sqrt{N} \rfloor - 1$ , so by definition  $0 < P_0 < \sqrt{N}$ .

Since  $x_0$  meets (A.4),  $|\sqrt{N} - Q_0| < P_{-1}$ .

Induction: Assume  $|\sqrt{N} - Q_i| < P_{i-1}$  and  $0 < P_{i-1} < \sqrt{N}$ .

Note that these assumptions require that  $0 < Q_i < 2\sqrt{N}$ . From (c),  $0 < x_i - b_i < 1$  means  $0 < \frac{\sqrt{N} - P_i}{Q_i} < 1$ . Since  $Q_i > 0$ ,  $0 < \sqrt{N} - P_i < Q_i$ . From the left side of this,  $P_i < \sqrt{N}$ . Now, either  $Q_i \leq \sqrt{N}$  or  $Q_i > \sqrt{N}$ .

Case 1: If  $Q_i \leq \sqrt{N}$ , then  $\sqrt{N} - P_i < Q_i \leq \sqrt{N}$ , so that  $P_i > 0$ .

Case 2: If  $Q_i > \sqrt{N}$ , then by (b),  $P_i = b_i Q_i - P_{i-1} > b_i \sqrt{N} - \sqrt{N} = (b_i - 1)\sqrt{N} \geq 0$ .

Therefore,  $0 < P_i < \sqrt{N}$ .

Since  $x_{i+1} > 1$ , it is trivial that  $Q_{i+1} < \sqrt{N} + P_i$  so that showing  $|\sqrt{N} - Q_{i+1}| < P_i$  reduces to showing  $Q_{i+1} > \sqrt{N} - P_i$ . Since  $1 = \frac{N - P_i^2}{Q_i Q_{i+1}} = \frac{\sqrt{N} + P_i}{Q_i} \frac{\sqrt{N} - P_i}{Q_{i+1}}$ , this is equivalent to showing:

$$\frac{\sqrt{N} + P_i}{Q_i} > 1. \tag{A.6}$$

Assume the contrary, that  $Q_i \geq \sqrt{N} + P_i$ . Then,

$$b_i(\sqrt{N} + P_i) - P_i \leq b_i Q_i - P_i = P_{i-1} < \sqrt{N},$$

$$b_i \sqrt{N} + P_i(b_i - 1) < \sqrt{N},$$

$$\sqrt{N}(b_i - 1) + P_i(b_i - 1) < 0,$$

$$(b_i - 1)(\sqrt{N} + P_i) < 0.$$

But  $\sqrt{N}$  and  $P_i$  are positive, so this implies  $b_i < 1$ , contradicting Theorem 2 (c). Therefore,

(A.6) holds.

(f) The fact that  $N = P_i^2 + Q_i Q_{i+1}$  requires that  $Q_{i+1} = \frac{N-P_i^2}{Q_i}$ . In order to show that  $\forall i$   $Q_i$  is an integer, the statements that  $Q_i$  is an integer and  $Q_i \mid N - P_i^2$  may be proven inductively.

Base case:  $i = 0$

By definition,  $Q_0$  is an integer and  $P_{-1}^2 \equiv N \pmod{Q_0}$ . But  $P_0 \equiv P_{-1} \pmod{Q_0}$ , so  $P_0^2 \equiv N \pmod{Q_0}$ . Therefore,  $Q_0 \mid N - P_0^2$ .

Induction: Assume for some  $i$ ,  $Q_i$  is an integer and  $Q_i \mid (N - P_i^2)$ . Then, since  $N = P_i^2 + Q_i Q_{i+1}$ ,  $Q_{i+1} = \frac{N-P_i^2}{Q_i}$ , so that since  $Q_i \mid (N - P_i^2)$ ,  $Q_{i+1}$  is an integer. Also,  $Q_i = \frac{N-P_i^2}{Q_{i+1}}$ , so that since  $Q_i$  is an integer,  $Q_{i+1} \mid (N - P_i^2)$ , so that  $P_i^2 \equiv N \pmod{Q_{i+1}}$ . Since  $P_{i+1} = b_{i+1} Q_{i+1} - P_i$ ,  $P_{i+1} \equiv P_i \pmod{Q_{i+1}}$ . Therefore,  $P_{i+1}^2 \equiv N \pmod{Q_{i+1}}$ , so that  $Q_{i+1} \mid (N - P_{i+1}^2)$  and the induction is complete.

(g) Solving (b) for  $b_i$  gives  $\frac{P_{i-1}+P_i}{Q_i} = b_i$ . Multiply by  $(P_{i-1} - P_i)$  to obtain:

$$\frac{P_{i-1}^2 - P_i^2}{Q_i} = b_i(P_{i-1} - P_i)$$

Rearranging and adding  $\frac{N}{Q_i}$  gives:

$$\begin{aligned} \frac{N - P_i^2}{Q_i} &= \frac{N - P_{i-1}^2}{Q_i} + b_i(P_{i-1} - P_i) \\ Q_{i+1} &= Q_{i-1} + b_i(P_{i-1} - P_i) \end{aligned}$$

(h) Since each  $x_i$  and thus the entire sequence that follows it is defined by the two integers  $Q_i$  and  $P_{i-1}$ , limited by the bounds  $0 < Q_i < 2\sqrt{N}$  and  $0 < P_i < \sqrt{N}$ , there is only a finite number of distinct  $x_i$ 's. Therefore, for some  $\pi$  and some  $k$ ,  $\forall i \geq k$   $x_i = x_{i+\pi}$ . **QED**

The fact that each  $x_i$  reduces to the form  $\frac{\sqrt{N}+P_{i-1}}{Q_i}$  is important for computational efficiency because this together with (c) imply that floating point arithmetic is not necessary for any of these calculations. Also, by use of (b) and (g), the arithmetic used in this recursion is on integers  $< 2\sqrt{N}$ .

One application of continued fractions is rational approximations.

$$\sqrt{N} = b_0 + \frac{1}{b_1 + \frac{1}{b_2 + \dots}}$$

If this continued fraction is truncated at any point, the result is an approximation to  $\sqrt{N}$ . One might imagine that it is necessary to start simplifying at the lower right end of this expression to obtain this approximation. However, Theorem 3, also included in [23], provides a simpler answer.

**Theorem 3** *Let:*

$$A_{-1} = 1, \quad A_0 = b_0, \quad A_i = b_i A_{i-1} + A_{i-2}, \quad i > 0$$

$$B_{-1} = 0, B_0 = 1, B_i = b_i B_{i-1} + B_{i-2}, i > 0$$

Then for  $i \geq 0$ ,  $[b_0, b_1, \dots, b_i] = \frac{A_i}{B_i}$  and  $A_{i-1}^2 - B_{i-1}^2 N = (-1)^i Q_i$ .

Note that the last equation gives  $A_{i-1}^2 \equiv (-1)^i Q_i \pmod{N}$ . Although this equation will change some when generalized to other continued fractions, these denominators  $\{Q_i\}$  will consistently be referred to as *pseudo-squares*. A proof of this theorem is given in [23]. Therefore, instead of reproducing the proof, I will provide an example:

### Example 10

$$\begin{aligned} x_0 &= \sqrt{403} + 20 = 40 + \sqrt{403} - 20 \\ x_1 &= \frac{1}{\sqrt{403}-20} = \frac{\sqrt{403}+20}{3} = 13 + \frac{\sqrt{403}-19}{3} \\ x_2 &= \frac{3}{\sqrt{403}-19} = \frac{\sqrt{403}+19}{14} = 2 + \frac{\sqrt{403}-9}{14} \\ x_3 &= \frac{14}{\sqrt{403}-9} = \frac{\sqrt{403}+9}{23} = 1 + \frac{\sqrt{403}-14}{23} \\ x_4 &= \frac{23}{\sqrt{403}-14} = \frac{\sqrt{403}+14}{9} = 3 + \frac{\sqrt{403}-13}{9} \\ x_5 &= \frac{9}{\sqrt{403}-13} = \frac{\sqrt{403}+13}{26} = 1 + \frac{\sqrt{403}-13}{26} \\ x_6 &= \frac{26}{\sqrt{403}-13} = \frac{\sqrt{403}+13}{9} = 3 + \frac{\sqrt{403}-14}{9} \\ x_7 &= \frac{9}{\sqrt{403}-14} = \frac{\sqrt{403}+14}{23} = 1 + \frac{\sqrt{403}-9}{23} \end{aligned}$$

From this, a table may be used to recursively calculate the approximation<sup>2</sup>:

$i$	-1	0	1	2	3	4	5	6
$b_i$		20	13	2	1	3	1	3
$A_i$	1	20	261	542	803	2951	3754	14213
$B_i$	0	1	13	27	40	147	187	708

filling in  $A_i$  and  $B_i$  from left to right. From the last column,  $\sqrt{403} \approx 14213/708$ .

Since the continued fraction is eventually periodic, it is reasonable to consider that when it loops around on itself, the terms being considered may have come from some terms “earlier” in the recursion. Example 10 provides some indication as to how the recursive formulas may be reversed, as  $\{Q_i\}$  and  $\{b_i\}$  are symmetric about  $x_5$ , so that after  $x_5$  these numbers are cycled through in reverse order. Lemma 1 addresses how each  $b_i$  is calculated two different ways and Lemma 2 shows that by exchanging these two related expressions, the direction is reversed.

### Lemma 1

$$\lfloor \frac{\sqrt{N} + P_i}{Q_i} \rfloor = \lfloor \frac{\sqrt{N} + P_{i-1}}{Q_i} \rfloor = b_i$$

---

<sup>2</sup>Actually, in this case  $b_0 = 40$ , but since that would be an approximation to  $x_0 = \sqrt{403} + 20$ , subtracting 20 from  $b_0$  yields an approximation to  $\sqrt{403}$ .

**Proof:** The second part of this equation, that  $\lfloor \frac{\sqrt{N}+P_{i-1}}{Q_i} \rfloor = b_i$  follows from the definition of  $b_i$ .

Theorem 2 (e) implies that  $Q_i > \sqrt{N} - P_{i-1}$ . Therefore,

$$\lfloor \frac{\sqrt{N} + P_i}{Q_i} \rfloor = \lfloor \frac{\sqrt{N} + b_i Q_i - P_{i-1}}{Q_i} \rfloor = b_i + \lfloor \frac{\sqrt{N} - P_{i-1}}{Q_i} \rfloor = b_i. \text{ QED}$$

Considering Example 10, it is then natural to suspect that the mechanism for going in the opposite direction will be precisely the same as the standard approach, except that the numerator is changed first. Note that this same change (with the exception of  $c_0$ ) could be achieved by merely changing the sign of  $P_{i-1}$ .

**Lemma 2** Let  $x_i, b_i, P_i, Q_i$ , and  $N$  be as in Theorem 2,  $i \geq 0$ . Let  $y_0 = \frac{\sqrt{N}+P_{i+1}}{Q_{i+1}}$  and let  $c_0 = \lfloor y_0 \rfloor$ . Define inductively  $y_j = \frac{1}{y_{j-1} - c_{j-1}}$ . Then  $c_0 = b_{i+1}$  and  $y_j = \frac{\sqrt{N}+P_{i-j+1}}{Q_{i-j+1}}$ ,  $j \geq 0$ .

**Proof:** By (A.6) and Lemma 1,  $c_0 = \lfloor y_0 \rfloor = \lfloor \frac{\sqrt{N}+P_{i+1}}{Q_{i+1}} \rfloor = b_{i+1}$ . By mathematical induction it suffices to prove the case  $j = 1$ . Using Theorem 2

$$\begin{aligned} y_1 &= \frac{1}{y_0 - c_0} = \frac{1}{\frac{\sqrt{N}+P_{i+1}}{Q_{i+1}} - b_{i+1}} = \frac{1}{\frac{\sqrt{N}+P_{i+1}-b_{i+1}Q_{i+1}}{Q_{i+1}}} \\ &= \frac{1}{\frac{\sqrt{N}-P_i}{Q_{i+1}}} = \frac{\sqrt{N}+P_i}{\frac{N-P_i^2}{Q_{i+1}}} = \frac{\sqrt{N}+P_i}{Q_i} \text{ QED} \end{aligned}$$

This demonstrates an important fact about continued fractions, the fact that the direction of the sequences of pseudo-squares and residues can be reversed (i.e. the indices decrease) by making a slight change and applying the same recursive mechanism.

Using Lemma 2,  $x_3$  may be used, for example, to find  $x_2$  and  $x_1$ . Continuing this process, denote the terms before  $x_0$  as  $x_{-1}, x_{-2}, \dots$ . Define  $Q_{-i}$  and  $P_{-i}$  similarly<sup>3</sup>. Example 11 demonstrates this with the continued fractions from Example 10:

**Example 11**  $x_3 = \frac{\sqrt{403}+9}{23}$  and  $P_3 = 14$ , so let  $y_0 = \frac{\sqrt{403}+14}{23}$  to obtain

$$\begin{aligned} y_0 &= \frac{\sqrt{403}+14}{23} = 1 + \frac{\sqrt{403}-9}{23} \\ y_1 &= \frac{23}{\sqrt{403}-9} = \frac{\sqrt{403}+9}{14} = 2 + \frac{\sqrt{403}-19}{14} \\ y_2 &= \frac{14}{\sqrt{403}-19} = \frac{\sqrt{403}+19}{3} = 13 + \frac{\sqrt{403}-20}{3} \\ y_3 &= \frac{3}{\sqrt{403}-20} = \frac{\sqrt{403}+20}{1} = 40 + \frac{\sqrt{403}-20}{1} \\ y_4 &= \frac{1}{\sqrt{403}-20} = \frac{\sqrt{403}+20}{3} = 13 + \frac{\sqrt{403}-19}{3} \\ y_5 &= \frac{3}{\sqrt{403}-19} = \frac{\sqrt{403}+19}{14} = 2 + \frac{\sqrt{403}-9}{14} \end{aligned}$$

---

<sup>3</sup>Since  $y_0$  meets (A.4) and the same recursive formula is applied, it is clear that Theorem 2 still applies to negative indices.

Then, just as  $y_2$  gives  $x_1 = \frac{\sqrt{403}+20}{3}$ ,  $y_4$  gives  $x_{-1} = \frac{\sqrt{403}+19}{3}$  and  $y_5$  gives  $x_{-2} = \frac{\sqrt{403}+9}{14}$ .

Combining periodicity with reversibility strengthens Theorem 2 (h).

**Lemma 3** *There exists a positive integer  $\pi$  such that  $\forall i$   $x_i = x_{i+\pi}$ ,  $i$  not necessarily positive.*

**Proof:** From the proof of Theorem 2 (h) there are  $k$  and  $\pi$  such that  $\forall i \geq k$ ,  $x_i = x_{i+\pi}$ . Essentially, this is equivalent to proving that there is no lower bound for  $k$ . Assume the contrary, that there is some lower bound  $k$ . Let  $k$  and  $\pi$  be the smallest such integers. Then  $x_k = x_{k+\pi}$ . But by Lemma 2  $x_{k-1} = x_{k+\pi-1}$ , so that  $k-1$  also meets this criteria, violating the assumption that  $k$  is the smallest such integer. Therefore,  $\forall i$   $x_i = x_{i+\pi}$ . **QED**

Throughout,  $\pi$  will consistently denote the period, even when considering this period in the context of quadratic forms or lattices.

Often the continued fraction may have other characteristics that are interesting besides its periodicity. For factorization, continued fractions with symmetries, such as at  $x_0$  and  $x_5$  from Example 10, will be especially important. If the starting condition near some point is the same in both directions, the entire sequence will be symmetric about that point. This is the point of Lemma 4.

**Lemma 4** *Let  $x_0 = \frac{\sqrt{N}+P_{-1}}{Q_0}$  meet (A.4) such that  $Q_0 \mid 2P_{-1}$ . The sequence of pseudo-squares is symmetric about  $Q_0$ , so that  $\forall i$   $Q_i = Q_{-i}$ .*

**Proof:**

Observe that  $0 < \sqrt{N} - P_0 < Q_0$ , with  $P_0 = b_0 Q_0 - P_{-1}$ , so that

$$0 < \sqrt{N} - b_0 Q_0 + P_{-1} < Q_0.$$

There can only be one possible integer value of  $b_0$  that satisfies this inequality. Since  $0 < \sqrt{N} - P_{-1} < Q_0$ ,  $b_0 = 2P_{-1}/Q_0$  satisfies this inequality, so that  $P_0 = P_{-1}$ .

Let  $y_{-1} = \frac{\sqrt{N}+P_1}{Q_1}$ . Then, by Lemma 2,  $y_0 = \frac{\sqrt{N}+P_0}{Q_0} = \frac{\sqrt{N}+P_{-1}}{Q_0} = x_0$

Therefore, the sequence of pseudo-squares will be symmetric about  $Q_0$ , since in either direction the first continued fraction term is the same. Therefore,  $Q_i = Q_{-i}$ . **QED**

The presence of one point of symmetry allows a proof that another point of symmetry exists and that a factorization of  $N$  may be obtained from this symmetry<sup>4</sup>:

**Theorem 4** *Let  $s = \lfloor \frac{\pi}{2} \rfloor$ , where  $\pi$  is the period from Lemma 3. If  $\pi$  is even,  $\forall i$   $Q_{s+i} = Q_{s-i}$ , but  $Q_s \neq Q_0$  and  $Q_s \mid 2N$ . If  $\pi$  is odd,  $\forall i$   $Q_{s+i+1} = Q_{s-i}$  and either  $\gcd(Q_s, N)$  is a nontrivial factor of  $N$  or  $-1$  is a quadratic residue of  $N$ .*

<sup>4</sup>This was actually discovered in the opposite order. It was clear that ambiguous forms that met this criteria provided a factorization but was later realized that these same forms produced symmetry points. This was first noticed by Gauss [7] and first applied by Shanks [29].

**Proof:**

Case 1: If  $\pi$  is even,  $\pi = 2s$ . Then, by Lemmas 4 and 3,  $Q_{s+i} = Q_{-s-i} = Q_{2s-s-i} = Q_{s-i}$ . Since  $Q_{s+1} = \frac{N-P_s^2}{Q_s}$  and  $Q_{s-1} = \frac{N-P_{s-1}^2}{Q_s}$ , this simplifies to  $P_s^2 = P_{s-1}^2$ , but since  $\forall i P_i > 0$ , this provides  $P_s = P_{s-1}$ .

Now  $Q_s = \frac{P_s+P_{s-1}}{b_s} = \frac{2P_s}{b_s}$ , so that  $Q_s \mid 2P_s$ .

Assume  $Q_s = Q_0$ . If  $Q_s$  is even, then  $P_0 \equiv P_s \equiv 1 \pmod{2}$  and if  $Q_s$  is odd,  $Q_s \mid P_s$ . Either way, there is then a unique integer in the range  $(\sqrt{N} - Q_0, \sqrt{N})$  satisfying these conditions, so that  $P_s = P_0$ . Therefore,  $x_s = \frac{\sqrt{N}+P_0}{Q_0} = x_0$ , contradicting the fact that  $\pi$  is the smallest positive integer such that  $\forall i Q_i = Q_{i+\pi}$ . Therefore,  $Q_s \neq Q_0$ .

Now  $N = P_s^2 + Q_s Q_{s+1}$ , so it is apparent that if  $Q_s$  is odd, then  $Q_s \mid P_s$ , so that  $Q_s \mid N$ . Conversely, if  $Q_s$  is even, then  $(Q_s/2) \mid P_s$ , so that  $(Q_s/2) \mid N$ . Either way,  $Q_s \mid 2N$ .

Case 2: If  $\pi$  is odd,  $\pi = 2s + 1$ . Then, by Lemma 4 and 3,  $Q_{s+i+1} = Q_{-s-i-1} = Q_{2s+1-s-i-1} = Q_{s-i}$ .

Specifically,  $Q_s = Q_{s+1}$ , so that  $N = P_s^2 + Q_s Q_{s+1} = P_s^2 + Q_s^2$ , so that  $P_{s+1}^2 \equiv -Q_s^2 \pmod{N}$ . If  $\gcd(Q_s, N) > 1$ , this is a nontrivial factor of  $N$ , and the proof is done. Therefore, assume that  $Q_s$  and  $N$  are relatively prime, so that  $Q_s^{-1} \pmod{N}$  exists. Then  $(Q_s^{-1})^2 P_{s+1}^2 \equiv -1 \pmod{N}$ . Then  $Q_s^{-1} P_{s+1}$  is a square root of  $-1$  modulo  $N$ . **QED**

One final concept that will appear much more important in later sections is equivalence. Define the set  $\mathbb{T}$  to be set of all numbers of the form  $\frac{\sqrt{N}+P}{Q}$  such that:

$$P^2 \equiv N \pmod{Q}, \tag{A.7}$$

Then define

$$\mathbb{T}^* = \{x \in \mathbb{T} : 0 < P < \sqrt{N}, |\sqrt{N} - Q| < P\}.$$

An element  $x \in \mathbb{T}$  is *reduced* if  $x \in \mathbb{T}^*$ . For  $x, y \in \mathbb{T}^*$ ,  $x$  is *equivalent* to  $y$  if  $x$  appears in the same continued fraction expansion as  $y$  and it is trivial that this is an equivalence relation on  $\mathbb{T}^*$ . Extending this to all of  $\mathbb{T}$  requires a lemma relating elements of  $\mathbb{T} - \mathbb{T}^*$  with elements of  $\mathbb{T}^*$ .

**Lemma 5** *Let  $x = x_0 \in \mathbb{T} - \mathbb{T}^*$ .  $x_0$  may be reduced by applying*

$$x_{i+1} = \frac{1}{x_i - b_i}, \quad b_i = \lfloor x_i - 1/2 \rfloor, \quad i \geq 0 \tag{A.8}$$

*until  $|Q_i| < 2\sqrt{N}$  for some  $i$  and then applying equation (A.2) normally until  $x_k \in \mathbb{T}^*$  for some  $k > 0$ .*

**Proof:** The choice of  $b_i$  yields that  $\left| \frac{\sqrt{N}-P_i}{Q_i} \right| < \frac{1}{2}$  after the first step, so that  $|P_i| < \frac{1}{2}|Q_i| + \sqrt{N}$ . Therefore,

$$\begin{aligned}
|Q_{i+1}| &= \left| \frac{N-P_i^2}{Q_i} \right| \\
&= \left| \frac{\sqrt{N}-P_i}{Q_i} \right| |\sqrt{N} + P_i| \\
&< \frac{1}{2} (2\sqrt{N} + \frac{1}{2}|Q_i|) \\
&= \sqrt{N} + \frac{1}{4}|Q_i|
\end{aligned}$$

so that  $|Q_i|$  will decrease as long as  $|Q_i| > \frac{4}{3}\sqrt{N}$ . When  $|Q_r| < 2\sqrt{N}$ , revert back to the standard formula for  $b_r$ . There are three cases for what  $Q_r$  is:

Case 1:  $0 < Q_r < \sqrt{N}$ . In this case, it is clear that  $x_{r+1}$  will be reduced.

Case 2:  $\sqrt{N} < Q_r < 2\sqrt{N}$ . In this case, if  $P_r > 0$ , then  $x_{r+1}$  will be reduced. Otherwise,  $|P_r| < \sqrt{N}$ , so that  $x_{r+1}$  will be in Case 1.

Case 3:  $-2\sqrt{N} < Q_r < 0$ . Then  $\sqrt{N} < P_r < \sqrt{N} + |Q_r|$ , yielding  $Q_{r+1} < 2\sqrt{N} + |Q_r|$ . If  $Q_{r+1} < 2\sqrt{N}$ , it is in Case 1 or Case 2. If  $2\sqrt{N} < Q_{r+1} < 2\sqrt{N} + |Q_r|$ , the choice of  $b_r$  provides  $\sqrt{N} + P_r > Q_{r+1}$ , so that  $0 < P_{r+1} < \sqrt{N}$ , so that  $x_{r+2}$  will be in Case 1. **QED**

I will provide one example of reduction:

### Example 12

$$\begin{aligned}
x_0 &= \frac{\sqrt{403}+267}{-134} = -2 + \frac{\sqrt{403}-1}{-134} \\
x_1 &= \frac{-134}{\sqrt{403}-1} = \frac{\sqrt{403}+1}{-3} = -8 + \frac{\sqrt{403}-23}{-3} \\
x_2 &= \frac{-3}{\sqrt{403}-23} = \frac{\sqrt{403}+23}{42} = 1 + \frac{\sqrt{403}-19}{42} \\
x_3 &= \frac{42}{\sqrt{403}-19} = \frac{\sqrt{403}+19}{1} = 39 + \frac{\sqrt{403}-20}{1}
\end{aligned}$$

Lemma 5 defines a map from  $\mathbb{T}$  to  $\mathbb{T}^*$  (Elements of  $\mathbb{T}^*$  are mapped to themselves). Then two elements are equivalent if their corresponding elements of  $\mathbb{T}^*$  are equivalent and it is clear that this is still an equivalence relation. Essentially, this equates to saying that two numbers  $x$  and  $y$  are equivalent if their continued fraction expansions have the same “tail”, so that after a certain number of terms of each they have identical cycles. §4 will define a different equivalence relation on binary quadratic forms and then prove that it corresponds to this.

## 3 From Morrison-Brillhart to Shanks

Morrison and Brillhart developed one simple and fairly intuitive algorithm for using continued fractions for factorization [19]. The entire algorithm is a bit more complicated, but here is a description sufficient for our purposes.

If the equation

$$x^2 \equiv y^2 \pmod{N} \tag{A.9}$$

can be solved such that

$$x \not\equiv \pm y \pmod{N}, \tag{A.10}$$

then it is evident that  $\gcd(x - y, N)$  provides a nontrivial factor of  $N$ . Choosing a value for  $x$  and then looking for a value that works for  $y$  is not computationally effective for large  $N$ . Fermat, the first to employ this concept, tested values of  $x$  greater than  $\sqrt{N}$  to find a value such that  $x^2 \pmod{N}$  was already a perfect square. However, these numbers get large quickly. Continued fractions provide a better approach to achieving this and since  $0 < Q_i < 2\sqrt{N}$ , the chances of finding a perfect square are greatly improved. The second part of Theorem 3 states that  $A_{i-1}^2 - B_{i-1}^2 N = (-1)^i Q_i$ . Therefore  $A_{i-1}^2 \equiv (-1)^i Q_i \pmod{N}$ . If for some  $i$ ,  $Q_{2i}$  is a perfect square then this provides a solution to (A.9) and it only remains to check whether or not  $\gcd(x + y, N)$  or  $\gcd(x - y, N)$  provide a nontrivial factor of  $N$ . However, there are not very many perfect squares in the continued fraction expansion so Morrison and Brillhart [19] used products to obtain squares. For example, for  $N = 1333$ , the continued fraction expansion provides  $73^2 \equiv -3 \pmod{N}$  and  $1789^2 \equiv -12 \pmod{N}$ . From this,  $(73 \cdot 1789)^2 \equiv (-3)(-12) = 6^2 \pmod{N}$ , quickly yielding  $1333 = 31 \cdot 43$ .

There are a couple of problems with this algorithm. First, it requires the calculation of the  $A_i$ 's, which are of the same size as  $N$ , after reduction modulo  $N$ , while the rest of the algorithm only requires arithmetic on numbers of size  $\sqrt{N}$ . Second, after going through a nontrivial amount of computation to find a relation that solves (A.9), not all of these result in a factorization. I provide one example:

**Example 13** *In the continued fraction for  $\sqrt{1333}$ ,  $Q_6 = 9 = 3^2$  and  $A_5 = 10661$ , so that  $10661^2 \equiv 3^2 \pmod{1333}$ . Unfortunately  $10661 \equiv -3 \pmod{1333}$ , so this does not result in a nontrivial factor of 1333.*

Although it is well enough on a computer to run the algorithm a few times and have it fail a few times, Daniel Shanks decided he needed to understand it a little better. Based on an understanding of quadratic forms and the class group infrastructure, Daniel Shanks developed several very interesting algorithms for factorization ([29],[26]). First he developed an improvement to the Morrison-Brillhart algorithm. Roughly speaking, rather than saving the  $A_i$ 's, he was able to use composition of quadratic forms to combine numbers to produce squares and then use the “infrastructure” to use those squares to find a factorization. In addition, he developed from the concept of infrastructure a system of predicting whether or not any given square would provide a nontrivial factor. Unfortunately, this didn't save very much time and was a much more complicated algorithm than the Morrison-Brillhart algorithm.

From here the development of the algorithm was prompted by the number  $2^{60} + 2^{30} - 1$ . It failed a Fermat primality test<sup>5</sup>, but when Morrison and Brillhart tried to factor it, it failed 114 times. Therefore, they stopped, multiplied it by some small constant and tried again. This time it worked on the first try, but they wanted to know why it had failed so many times. So they asked Shanks to analyze it. Unfortunately (or fortunately in hindsight), Shanks only had an HP-65 available and he couldn't fit his entire algorithm into it. Therefore,

---

<sup>5</sup>The Fermat primality test is based on Fermat's classical result that for  $p$  prime,  $a^p \equiv a \pmod{p}$  [9]. Equivalently, if  $a^N \not\equiv a \pmod{N}$  for some integer  $a$ , then  $N$  isn't prime.

he discarded all the work of combining numbers to form squares and just cycled through until he found one already there. The code for this was much shorter, and as it turned out the algorithm was actually significantly faster.

## 4 Quadratic Forms

A fuller account of binary quadratic forms can be found in Gauss's [7] and in Buell's [2]. However, here are the necessary fundamental ideas.

A binary quadratic form is a polynomial of the form  $F(x, y) = ax^2 + bxy + cy^2$ ,  $x, y, a, b, c \in \mathbb{Z}$  (Often this is abbreviated as  $(a, b, c)$ ). In some sense then, a quadratic form may be considered to be the set of all the numbers it can represent for various values of  $x$  and  $y$ . Thus, two quadratic forms are *equivalent* if they represent the same set of integers. It is evident that if one form is transformed into another by the substitution

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix}, \quad ad - bc = \pm 1, \quad (\text{A.11})$$

then, since this matrix is invertible, the two forms are equivalent. As one further useful distinction, (A.11) is *proper* if its determinant is  $+1$  and *improper* if its determinant is  $-1$ . The symbol  $(\sim)$  will only apply to proper equivalence. For the purpose of factorization, the interesting forms are those that can be improperly transformed into themselves, referred to as *ambiguous forms*.

**Example 14**  $F(x, y) = -14x^2 + 10xy + 5y^2$  is transformed into itself by the substitution:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -2 & -1 \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix},$$

so  $(-14, 10, 5)$  is *ambiguous*.

Denote the set of all quadratic forms with discriminant  $\Delta$  by  $\mathbb{F}_\Delta$ , or often just  $\mathbb{F}$ . The next obvious question is the organization of all of the quadratic forms equivalent to some given form. Since there are an infinite number of forms equivalent to any form, the search must be narrowed some by first defining reduced forms.

**Definition 2** A quadratic form  $ax^2 + bxy + cy^2$ , with positive discriminant  $\Delta = b^2 - 4ac$  is *reduced* if:

$$0 < b < \sqrt{\Delta} \quad (\text{A.12})$$

$$|\sqrt{\Delta} - 2|a|| < b \quad (\text{A.13})$$

Note that  $\Delta = b^2 - 4ac$  and (A.12) require that  $ac < 0$ , so that  $a$  and  $c$  must have opposite signs.

Making Gauss's description of the organization make some sense will require one more of his definitions:

**Definition 3** Two forms  $F(x, y) = ax^2 + bxy + cy^2$  and  $F'(x, y) = a'x^2 + b'xy + c'y^2$  are *adjacent* if  $c = a'$ .

To each quadratic form, there is a unique reduced equivalent form adjacent to it on each side<sup>6</sup>, and since (A.12-A.13) imply a finite number of possible coefficients, this process eventually repeats, forming a cycle. The important aspect of this is that the cycle is actually all of the reduced forms equivalent to the first form:

**Theorem 5** [7] *If the reduced forms  $F, F'$  are properly equivalent, each of them will be contained in the period of the other.*

Gauss proves this in Article 193 of [7], Lenstra proves this in [15], and it is a corollary of Lemma 13 in §6. Therefore the proof is omitted.

For now, the important detail is that the quadratic forms correspond directly to the elements of  $\mathbb{T}$ , and that the reduced quadratic forms correspond to elements of  $\mathbb{T}^*$ . Note that the elements of  $\mathbb{T}$  have attached indices, where the important trait of the indice is whether it is odd or even. Define a map from  $\mathbb{T}$  to  $\mathbb{F}$  by

$$\begin{aligned} \Phi_{\mathbb{T}, \mathbb{F}} : \mathbb{T} &\rightarrow \mathbb{F} \\ \frac{\sqrt{N-P_i}}{Q_i} &\rightarrow F_i(x, y) = Q_i(-1)^i x^2 + 2P_i xy + Q_{i+1}(-1)^{i+1} y^2 \end{aligned} \quad (\text{A.14})$$

The inverse map is

$$\begin{aligned} \Phi_{\mathbb{F}, \mathbb{T}} : \mathbb{F} &\rightarrow \mathbb{T} \\ ax^2 + bxy + cy^2 &\rightarrow x_i = \frac{\sqrt{\Delta/4 - b/2}}{|a|} \in \mathbb{T} \end{aligned} \quad (\text{A.14}')$$

where the discriminant of the quadratic form is  $\Delta$  is the element of  $\mathbb{T}$  is given either an even or odd indice as  $a$  is positive or negative, respectively. Note that  $\Delta = b^2 - 4ac$  gives  $4a \mid \Delta - b^2$ , so that  $x_i$  really is in  $\mathbb{T}$ .

§2 defined an equivalence on  $\mathbb{T}$  and suggested that it corresponded with an equivalence of binary quadratic forms. Theorem 6 formalizes this:

**Theorem 6** *Under the mapping  $\Phi_{\mathbb{T}, \mathbb{F}}$ , the equivalence classes of  $\mathbb{T}$  correspond to the equivalence classes of  $\mathbb{F}$ . That is, for  $x_i, x_j \in \mathbb{T}$  corresponding to  $F_i = \Phi_{\mathbb{T}, \mathbb{F}}(x_i), F_j = \Phi_{\mathbb{T}, \mathbb{F}}(x_j) \in \mathbb{F}$ , respectively,  $x_i \sim x_j$  if and only if  $F_i \sim F_j$ .*

**Proof:**

Let  $x_i \sim x_j$ . Since  $x_i$  and  $x_j$  must be in the same continued fraction expansion, assume without loss of generality that  $j = i + 1$ . The other cases may be easily derived from this case. Then the quadratic form related to  $x_i$  is given in (A.14). The substitution

---

<sup>6</sup>Although Gauss had a recursive mechanism for finding these, continued fractions provide a sufficient mechanism for this that will be defined momentarily. Note that reversal suddenly becomes trivial.

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & (-1)^i b_{i+1} \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix}$$

transforms  $F_i$  into

$$Q_{i+1}(-1)^{i+1}x^2 + 2P_{i+1}xy + Q_{i+2}(-1)^{i+2}y^2$$

Observe that this matrix has determinant 1, so that this equivalence is proper.

In order to prove the converse, that the  $x_i$ 's related to equivalent quadratic forms are equivalent, by Theorem 5, observe that the last coefficient of the quadratic form related to  $x_i$ ,  $Q_{i+1}(-1)^{i+1}$ , is then the first coefficient of the quadratic form related to  $x_{i+1}$ . Therefore, these two forms are adjacent and thus equivalent. **QED**

The real value of quadratic forms is the composition of quadratic forms. In Article 236 of [7], Gauss provides a very flexible definition of composition. Gauss defines composition as multiplying two quadratic forms together and then making a substitution to simplify this into another binary quadratic form. The algorithm he provides is very complicated, allowing for choices of variables along the way that permit the result to be any quadratic form in the resulting equivalence class. The result of composition should be predictable, so definition needs to be limited some. Shanks and Buell both provide a significant simplification of this algorithm. The symbol  $(*)$  will consistently be used for composition.

**Proposition 2** [2] *Let  $F_1 = (a_1, b_1, c_1)$  and  $F_2 = (a_2, b_2, c_2)$  be primitive forms of discriminants  $d_1$  and  $d_2$ , respectively, such that  $d_1 = \Delta n_1^2$  and  $d_2 = \Delta n_2^2$  for integers  $n_1$  and  $n_2$  and  $\Delta$ , with  $\Delta = \gcd(d_1, d_2)$ . Let*

$$m = \gcd(a_1 n_2, a_2 n_1, \frac{b_1 n_2 + b_2 n_1}{2}).$$

*Then the congruences*

$$\begin{aligned} mn_1 B &\equiv mb_1 \pmod{2a_1} \\ mn_2 B &\equiv mb_2 \pmod{2a_2} \\ m(b_1 n_2 + b_2 n_1) B &\equiv m(b_1 b_2 + \Delta n_1 n_2) \pmod{4a_1 a_2} \end{aligned}$$

*are simultaneously solvable<sup>7</sup> for an integer  $B$ , and the composition of  $F_1$  and  $F_2$  is:*

$$F_1 * F_2 = \left( \frac{a_1 a_2}{m^2}, B, \frac{(B^2 - \Delta)m^2}{4a_1 a_2} \right)$$

*of discriminant  $\Delta$ .*

See [28] for a derivation of this in the case where the discriminants are equal or [2] for a proof of this case. Buell [2] also provides the substitutions that would be needed for Gauss's definition of composition.

The next question is how this operation is related to equivalence.

---

<sup>7</sup>By convention, choose the answer with the smallest absolute value.

**Theorem 7** *If  $F_1 \sim F_2$ , then  $F * F_1 \sim F * F_2$ .*

Gauss proves this in Article 237-239 of [7].

Therefore, composition treats the equivalence classes in the convenient manner. These equivalence classes are then the elements of the class group, with composition as the group operation. The application of Theorem 7 is that it doesn't matter which form is used to represent an equivalence class.

The significance of ambiguous forms for factorization has been mentioned some above. It is evident that if one form is ambiguous, then its entire equivalence class is also ambiguous. Lemmas 6 generalizes the reasons to be interested in these classes.

**Lemma 6** *An ambiguous equivalence class contains two points of symmetry, that is, pairs of reduced adjacent forms,  $(c, b, a)$  and  $(a, b, c)$  in the cycle that are the symmetric reverse of each other. Let  $a$  be the connecting term of either symmetry point. Either  $a$  divides the determinant, or  $a/2$  divides the determinant.*

**Proof:**

Let  $A$  be an ambiguous equivalence class and let  $F = ax^2 + bxy + cy^2 \in A$ . Let  $F' = cx^2 + bxy + ay^2$ . Then since  $F \in A$ , there is a substitution of determinant  $-1$  that maps  $F$  into itself. Since the obvious substitution to exchange  $x$  and  $y$  in  $F$  has determinant  $-1$ , the product of these two is a proper substitution that transforms  $F$  into  $F'$ . Therefore,  $F' \in A$ , so that if  $F$  is  $F_0$ ,  $F'$  is  $F_j$  for some  $j$ . Then that  $F_1$  must be the reverse of  $F_{j-1}$ , and so forth. Now, if  $j$  is even, then by this process  $F_{j/2}$  is its own reverse. However, by the definition of being reduced, the end coefficients of each form must have opposite sign, so this is impossible. Therefore,  $j$  must be odd, and then  $F_{\frac{j-1}{2}}$  is the reverse of  $F_{\frac{j+1}{2}}$ .

At this point, observe that since the end-coefficients alternate signs, the entire period must be even. By the same arguments as Theorem 4, one could show that there must be another point of symmetry with the property that  $\forall i \ Q_{s+i} = Q_{s-i}$ , but such that  $Q_s$  is not the same as the connecting term at the first symmetry point. The two quadratic forms containing  $Q_s$  as an end coefficient then meet the criteria.

The fact that either  $a$  divides the determinant, or  $a/2$  divides the determinant was proven in Theorem 4, since the determinant is  $4N$ . **QED.**

Note that Theorem 4 described two different types of points of symmetry. With the quadratic form cycle, the second case can be ignored because of the alternating signs. However, it is quite possible for the term at one symmetry point to be merely the negative of the term at the other symmetry point. This would correspond to the continued fraction having an odd period and there would be a symmetry point of the second type in the continued fraction at half-way. However, this type of symmetry does not generally provide a factorization for  $N$ .

Lastly, it is important how these ambiguous forms fit into the rest of the class group. First, addressing the class group structure requires inverses. Lemma 7 is fairly elementary and is probably stated somewhere else. Let 1 represent the form in the principal cycle whose first coefficient is 1. Let  $F^{-1}$  indicate the symmetric reverse of  $F$ ,  $(a, b, c)^{-1} = (c, b, a)$ . Lemma 7 justifies this notation:

**Lemma 7**  $F * F^{-1} \sim 1$

**Proof:**

Let  $F = ax^2 + bxy + cy^2$ . Then  $F^{-1} = cx^2 + bxy + ay^2$ . Let  $G$  be the next form adjacent to  $F^{-1}$ , that is  $G = ax^2 + b'xy + c'y^2$ , with  $a \mid (b+b')$  from the correspondance with continued fractions. Composing  $F * G$ ,  $n_1 = n_2 = 1$  and  $m = a$ , so that the first coefficient of  $F * G$  is 1. Therefore,  $F * G \sim 1$ , but  $F^{-1} \sim G$ , so  $F * F^{-1} \sim 1$ . **QED.**

Note that this implies that the square of a symmetry point is 1.

Theorem 8 was probably known by Shanks, since SQUFOF depends highly on it, but it does not seem that he states this explicitly anywhere.

**Theorem 8** *An equivalence class has order 2 or 1 in the class group if and only if it is ambiguous.*

**Proof:**

Let  $A$  be an ambiguous class. Let  $F \in A$ . Then  $F \sim F^{-1}$ , so that  $F * F \sim F * F^{-1} \sim 1$ . Therefore  $F * F$  is in the principal cycle, so that  $A$  has order 2 or 1 in the class group.

Conversely, assume that an equivalence class  $A$  has order 2 or 1 in the class group. Let  $F \in A$ . Then  $F * F$  is in the principal ideal, so that  $F * F \sim (F * F)^{-1}$ . But from composition, it is clear that  $(F * F)^{-1} \sim F^{-1} * F^{-1}$ . So  $F * F \sim F^{-1} * F^{-1}$ . Since the class group is associative, composing on the right with  $F$  maintains equivalence. Therefore:

$$\begin{aligned} (F * F) * F &\sim (F^{-1} * F^{-1}) * F \\ 1 * F &\sim F^{-1} * (F^{-1} * F) \\ F &\sim F^{-1} \end{aligned}$$

Therefore,  $A$  is ambiguous. **QED.**

Certainly the class group structure is interesting, but it is now possible to return to the problem from the Morrison-Brillhart algorithm of Example 13 with  $Q_3 = 3$ , so  $Q_6 = 9$  doesn't provide a nontrivial factor of  $N$ . The quick explanation is that if you square the quadratic form with first coefficient  $Q_3$ , you obtain the quadratic form with first coefficient  $Q_6$ . Since the principal cycle is closed under composition, it seems as though, and perhaps would be convenient if, the forms in the principal cycle formed a group. However, the problem of reduction prevents this:

**Example 15** *Consider the quadratic form  $F = (36, 70, -3)$ , with determinant  $4 \cdot 1333$ . Compare  $(F * F) * F$ , with  $F * F * F$ , where the difference is that in the first the result is reduced after the first composition.  $F * F = (324, -38, -3)$  and the very next adjacent form  $(-3, 68, 59)$ , is reduced.  $F * (-3, 68, 59) = (-12, 70, 9)$ , which is already reduced. However, without reduction  $F * F * F = F * (324, -38, -3) = (729, 448, -348)$ . When this is reduced, the first reduced form found, after 2 steps, is  $(9, 56, -61)$ .*

Therefore, the principal cycle, with the operation being composition followed by reduction, doesn't even meet the requirements for being power associative. However, the observation

that the two results are adjacent forms, and that the second reduction took one step longer, prompts us to dig a little deeper.

Understanding this requires what Shanks referred to as infrastructure distance. For  $m < n$ , and for  $x_i \in \mathbb{T}$ , the terms in the continued fraction in (A.5), define

$$D_{\mathbb{F}}(x_m, x_n) = \log\left(\prod_{k=m+1}^n x_k\right) \quad (\text{A.15})$$

Lenstra [15] adds a term of  $\frac{1}{2} \log(Q_n/Q_m)$  to this, with the effect that the resulting formulas are slightly simplified but the proofs are more complicated and less intuitive. This definition is used by Williams in [32].

Since the quadratic forms are cyclic, in order for the distance between two forms to be measured consistently, it must be considered modulo the distance around the principal cycle.

**Definition 4** Let  $\pi$  be the period of the principal cycle. The *regulator*  $R$  of the class group is the distance around the principal cycle, that is,

$$R = D_{\mathbb{F}}(F_0, F_{\pi}) = D(1, F_{\pi})$$

Therefore, distance must be considered modulo  $R$ , so that  $D_{\mathbb{F}}$  is a map from pairs of forms to the interval  $[0, R) \in \mathbb{R}$ . The addition of two distances must be reduced modulo  $R$  as necessary.

Further analysis of this distance will require two more tools: ideals and lattices. In order to relate to continued fractions, the ideals will be in  $\mathbb{Z}[\sqrt{N}] = \{a + b\sqrt{N} : a, b \in \mathbb{Z}\}$  and the lattices will be in  $\mathbb{Q}(\sqrt{N}) = \{a + b\sqrt{N} : a, b \in \mathbb{Q}\}$  where  $N$  is a non-square positive integer.

Remark: The ideals in  $\mathbb{Z}[\sqrt{N}]$  typically correspond only to quadratic forms of discriminant  $4N$ . Note that if  $N \equiv 1 \pmod{4}$ , then  $\mathbb{Z}[\sqrt{N}]$  is not the ring of integers for  $\mathbb{Q}(\sqrt{N})$ . For  $N \equiv 1 \pmod{4}$ , an analysis of ideals in  $\mathbb{Z}[\frac{\sqrt{N}+1}{2}]$  is also interesting, but will be avoided in the interest of simplicity. Quadratic forms of discriminant  $N \equiv 1 \pmod{4}$  may be related to ideals in  $\mathbb{Z}[\sqrt{N}]$  via first multiplying by 2 to obtain quadratic forms of discriminant  $4N$ .

## 5 Ideals

For  $\xi \in \mathbb{Q}(\sqrt{N})$ , let  $\bar{\xi}$  refer to the *conjugate* of  $\xi$  (i.e.  $\overline{1 + \sqrt{3}} = 1 - \sqrt{3}$ ).

The *norm* of a number in  $\mathbb{Q}(\sqrt{N})$  is  $\mathcal{N}(\xi) = \xi\bar{\xi} \in \mathbb{Q}$ .

To simplify notation, the symbols  $H$ ,  $I$ ,  $J$ , and  $K$  will consistently be ideals,  $u$  and  $v$  will be elements of ideals,  $\alpha$  and  $\beta$  will be elements of  $\mathbb{Z}[\sqrt{N}]$ ,  $\xi$  and  $\zeta$  will be elements of  $\mathbb{Q}(\sqrt{N})$ , and  $\mathcal{L}$  will be a lattice.

Our definition of an ideal is the same as in any other commutative ring with identity:

**Definition 5** A subset  $I$  of a ring  $R$  is an *ideal* if for  $u, v \in I$ ,  $u \pm v \in I$  and for  $\alpha \in R$ ,  $u \cdot \alpha \in I$ , that is  $I$  is closed under addition and multiplication by an element of  $R$ . Define  $L(I)$  to be the least positive rational integer in  $I$ .

Describing ideals will require the notation for the lattice generated by a set. If

$$\alpha_1, \alpha_2, \dots, \alpha_k \in \mathbb{Z}[\sqrt{N}],$$

denote<sup>8</sup> the lattice generated by these as

$$[\alpha_1, \alpha_2, \dots, \alpha_k] = \left\{ \sum_{i=1}^k n_i \alpha_i : n_i \in \mathbb{Z} \right\} \quad (\text{A.16})$$

Lemma 8 identifies necessary and sufficient conditions for a set in  $\mathbb{Z}[\sqrt{N}]$  to be an ideal.

**Lemma 8** *For  $Q, s, N, P \in \mathbb{Z}$ ,  $N$  non-square and positive,  $[Q, s\sqrt{N} + P]$  is an ideal of the ring  $\mathbb{Z}[\sqrt{N}]$  if and only if  $sQ \mid \mathcal{N}(s\sqrt{N} + P)$ ,  $s \mid Q$ , and  $s \mid P$ .*

**Proof:** Assume that  $I = [Q, s\sqrt{N} + P]$  is an ideal of  $\mathbb{Z}[\sqrt{N}]$ . Then, choosing  $\alpha = P - s\sqrt{N}$ ,  $\mathcal{N}(s\sqrt{N} + P) \in I$ . Since this is an integer,  $Q \mid \mathcal{N}(s\sqrt{N} + P)$ . Choosing  $\alpha = \sqrt{N}$ ,  $Q\sqrt{N} \in I$ . Therefore,  $s \mid Q$ . Since  $Q \mid \mathcal{N}(s\sqrt{N} + P)$ , this also implies that  $s \mid P$ . Therefore,  $\alpha$  could have been chosen  $\alpha = P/s - \sqrt{N}$  so that  $Q \mid \mathcal{N}(s\sqrt{N} + P)/s$ , so that  $sQ \mid \mathcal{N}(s\sqrt{N} + P)$ .

Conversely, let  $I = [Q, s\sqrt{N} + P]$  and assume that  $sQ \mid \mathcal{N}(s\sqrt{N} + P)$ ,  $s \mid Q$ , and  $s \mid P$ . Closure under addition is trivial. To see that  $I$  is closed under multiplication by an element of  $\mathbb{Z}[\sqrt{N}]$ , one need only consider multiplication by 1 and  $\sqrt{N}$ , since they form a basis for  $\mathbb{Z}[\sqrt{N}]$ . Multiplication by 1 is trivial. For  $\sqrt{N}$ ,

$$Q\sqrt{N} = \frac{Q}{s}(s\sqrt{N} + P) - \frac{P}{s}Q$$

and  $Q/s$  and  $-P/s$  are integers. Also,

$$(s\sqrt{N} + P)\sqrt{N} = sN + P\sqrt{N} = \frac{P}{s}(s\sqrt{N} + P) + \left(\frac{-P^2 + s^2N}{sQ}\right)Q$$

and  $\frac{P}{s}$  and  $\left(\frac{-P^2 + s^2N}{sQ}\right)$  are integers. **QED**

If  $s = 1$ , an ideal is *primitive*. Since  $s \mid P$  and  $s \mid Q$ , ideals that are not primitive will often be written  $(s)[Q, \sqrt{N} + P]$ . Let  $\mathbb{I}$  be the set of all primitive ideals.

Represented in the form  $I = [Q, \sqrt{N} + P]$ , it is clear that  $|Q|$  is the smallest positive rational integer in  $I$ . Define

$$L(I) = \min\{I \cap \mathbb{Z}^+\} \quad (\text{A.17})$$

At this point, it is possible to define a correspondance between quadratic forms (of discriminant  $\Delta \equiv 0 \pmod{4}$ ) and ideals by:

---

<sup>8</sup>Observe the difference between the use of [...] here and in §2. This expression is completely unrelated to rational approximations.

$$\Phi_{\mathbb{F},\mathbb{I}}(F(x, y) = Ax^2 + Bxy + Cy^2) = [A, \sqrt{(\frac{B}{2})^2 - AC} + \frac{B}{2}] \quad (\text{A.18})$$

$$\Phi_{\mathbb{I},\mathbb{F}}([Q, \sqrt{N} + P]) = F(x, y) = Qx^2 + 2Pxy + (\frac{P^2 - N}{Q})y^2 \quad (\text{A.18}')$$

and define a *reduced* ideal as an ideal corresponding to a reduced quadratic form. Note that  $\Delta = 4N$ .

For example, the quadratic form  $(15, 2 \cdot 12, -1)$  corresponds to the ideal  $[15, \sqrt{159} + 12]$ . The one potential problem that immediately becomes apparent is that while  $[15, \sqrt{159} + 12]$  and  $[-15, \sqrt{159} + 12]$  are the same ideal,  $(15, 2 \cdot 12, -1)$  and  $(-15, 2 \cdot 12, 1)$  are different quadratic forms. However, it is apparent that the negative sign is merely carried through composition without affecting the computations. Since each of these forms is in the same location within its respective cycle, this difference will not be important to this investigation of composition and distance.

$\Phi_{\mathbb{T},\mathbb{F}}$  and  $\Phi_{\mathbb{F},\mathbb{I}}$  may be combined to obtain

$$\Phi_{\mathbb{T},\mathbb{I}}(\frac{Q}{\sqrt{N} - P}) = [Q, \sqrt{N} - P]$$

and  $\Phi_{\mathbb{I},\mathbb{T}}$  is defined in the related obvious way.

If  $A = [\alpha_i]$  and  $B = [\beta_i]$ ,  $i = 1, 2, 3, \dots, d$ , then it is clear that  $A = B$  if and only if there exists a  $d \times d$  matrix  $M$  with determinant  $\pm 1$  such that:

$$\langle \alpha_i \rangle = M \langle \beta_i \rangle$$

where  $\langle \alpha_i \rangle$  and  $\langle \beta_i \rangle$  are vectors.

For these purposes, the most important operation with ideals is their multiplication. Multiplication is defined by

$$[\alpha_i] \cdot [\beta_j] = [\alpha_i \beta_j]$$

For example,

$$I = [15, \sqrt{159} + 12] \cdot [10, \sqrt{159} + 13] = [150, 10\sqrt{159} + 120, 15\sqrt{159} + 195, 315 + 25\sqrt{159}]$$

The 4th component is the sum of the 2nd and 3rd, so it is unnecessary for describing the ideal.

Applying the matrix

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix}$$

which has determinant 1, subtracts the 2nd component from the third to obtain

$$I = [150, 10\sqrt{159} + 120, 5\sqrt{159} + 75]$$

The matrix, with determinant 1,

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

will subtract twice the 3rd component from the 2nd to obtain

$$I = [150, -30, 5\sqrt{159} + 75]$$

Here the 1st component is a multiple of the 2nd and is thus unnecessary. The answer is simplified to obtain

$$I = [30, 5\sqrt{159} + 75] = 5[6, \sqrt{159} + 15]$$

The process of multiplying ideals can be greatly simplified by several well-known formulae<sup>9</sup> [18].

**Theorem 9** *Let  $I = [Q, \sqrt{N} + P]$  and  $J = [Q', \sqrt{N} + P']$  be ideals of  $Q\sqrt{N}$ . Let  $C = \frac{N-P^2}{Q}$ ,  $C' = \frac{N-(P')^2}{Q'}$ . If  $\gcd(Q, P, C) = \gcd(Q', P', C') = 1$ , then  $I \cdot J = s[q, \sqrt{N} + p]$ , where*

$$s = \gcd(Q, Q', P + P') \tag{A.19}$$

$$h = \gcd(Q, Q', C, C', 2) \tag{A.20}$$

$$q = hQQ'/s^2 \tag{A.21}$$

$$p \equiv P \pmod{Q/s} \tag{A.22}$$

$$p \equiv P' \pmod{Q'/s} \tag{A.23}$$

$$(P + P')p \equiv N + PP' \pmod{QQ'/s} \tag{A.24}$$

**Proof<sup>10</sup>:**

Consider the product:

$$I \cdot J = [QQ', Q\sqrt{N} + QP', Q'\sqrt{N} + Q'P, N + PP' + (P + P')\sqrt{N}] \tag{A.25}$$

The smallest integer in  $I \cdot J$  may be found by considering the smallest integers that may be produced taking these elements pair-wise.

---

<sup>9</sup>In [18], (A.24) is stated as  $(P - p)(P' - p) \equiv n + tp + p^2 \pmod{QQ'/s}$ , but in this case  $t = 0$  and  $n = -N$ .

<sup>10</sup>Some of the arguments were taken from Buell's proof in [2] concerning composition of quadratic forms.

$$L(I \cdot J) = \gcd(QQ', \text{lcm}(Q, Q')(P - P'), \frac{\text{lcm}(Q, P + P')Q'C'}{P + P'}, \frac{\text{lcm}(Q', P + P')QC}{P + P'})$$

Let  $s = \gcd(Q, Q', P + P')$ ,  $h = \gcd(Q, Q', C, C', 2)$ ,  $w = hQQ'/s$ . Let  $f \neq 2$  be a prime. Let  $a, b, c, d, e, k$  be the largest possible integers such that  $f^a \mid Q$ ,  $f^b \mid Q'$ ,  $f^c \mid (P + P')$ ,  $f^d \mid C$ ,  $f^e \mid C'$ ,  $f^k \mid (P - P')$ . Then  $f^{a+b} \parallel QQ'$ ,  $f^{\max(a,b)+k} \parallel \text{lcm}(Q, Q')(P - P')$ ,  $f^{\max(a,c)+b+e-c} \parallel \frac{\text{lcm}(Q, P+P')Q'C'}{P+P'}$ , and  $f^{\max(b,c)+a+d-c} \parallel \frac{\text{lcm}(Q', P+P')QC}{P+P'}$ .

The following analysis proves that if  $f \neq 2$ , the maximum exponent of  $f$  in  $L(I \cdot J)$  is  $a + b - \min(a, b, c)$  while if  $h = 2$ , then the maximum exponent of 2 in  $L(I \cdot J)$  is  $a + b + 1 - \min(a, b, c)$ , while if  $h = 1$ , then the maximum exponent of  $f$  in  $L(I \cdot J)$  is  $a + b - \min(a, b, c)$ . As this is broken in several different cases, an outline of the proof is helpful:

1)  $a = 0$  or  $b = 0$  or  $c = 0$

2)  $a \neq 0, b \neq 0$ , and  $c \neq 0$

2.1)  $f \neq 2$

2.1.1)  $a + d \neq b + e$

$$f \mid (P - P')$$

$$f \nmid (P - P')$$

2.1.2)  $a + d = b + e$

$$f \mid (P - P')$$

$$f \nmid (P - P')$$

2.2)  $f = 2$

2.2.1)  $a + d \neq b + e$

$$c > 1, k > 1$$

$$c > 1, k \leq 1$$

$$c = 1$$

2.2.2)  $a + d = b + e$

$$c > 1, k > 1$$

$$c = 1$$

$$k = 1$$

Case 1) If  $a = 0$ , then  $\max(a, c) + b + e - c = b + e \geq b$ ,  $\max(b, c) + a + d - c \geq b$  and  $a + b = b$ , so the maximum exponent for  $f$  in  $L(I \cdot J)$  is  $b = a + b - \min(a, b, c)$ . Similarly, if  $b = 0$ , then the maximum exponent is  $a = a + b - \min(a, b, c)$ .

Assume  $c = 0$ .  $f^{a+d} \parallel QC = N - P^2$  and  $f^{b+e} \parallel Q'C' = N - (P')^2$ , subtracting,  $f^{\min(a+d, b+e)} \mid (P^2 - (P')^2) = (P + P')(P - P')$ . Since  $c = 0$ , then  $f^{\min(a+d, b+e)} \mid (P - P')$ .

Therefore,  $f^{\max(a,b)+\min(a+d,b+e)} \mid \text{lcm}(Q, Q')(P - P')$ . However,  $\max(a, b) + \min(a + d, b + e) \geq \max(a, b) + \min(a, b) = a + b$ . Therefore, the maximum exponent for  $f$  in  $L(I \cdot J)$  is

$$\begin{aligned} \min(a + b, \max(a, c) + b + e - c, \max(b, c) + a + d - c) &= \min(a + b, a + b + e, a + b + d) \\ &= a + b = a + b - \min(a, b, c) \end{aligned}$$

Case 2.1.1) Assume  $c \neq 0$ ,  $f \neq 2$ , and  $a + d \neq b + e$ . Then,  $f^{\min(a+d,b+e)} \parallel (P^2 - (P')^2) = (P + P')(P - P')$ . If  $f \mid (P - P')$ , then  $f \mid 2P$  and  $f \mid 2P'$ . Since  $f \neq 2$ , this gives  $f \mid P$ ,  $f \mid P'$ . Then,  $d = e = 0$ . Also,  $c \leq \min(a, b)$ . Then, the maximum exponent for  $f$  in  $L(I \cdot J)$  is

$$\begin{aligned} \min(a + b, \max(a, b) + \min(a, b) - c, \max(a, c) + b - c, \max(b, c) + a - c) = \\ a + b - c = a + b - \min(a, b, c) \end{aligned}$$

If  $f \nmid (P - P')$  then  $c = \min(a + d, b + d)$  and the maximum exponent for  $f$  in  $L(I \cdot J)$  is

$$\min(a + b, \max(a, b), \max(a, c) + b + e - c, \max(b, c) + a + d - c)$$

If  $a = \min(a, b, c)$ , then this is  $\min(b, c + b + e - c, \max(b, c) + a + d - c) = b = a + b - \min(a, b, c)$ . The case is similar if  $b = \min(a, b, c)$ . If  $c = \min(a, b, c)$ , then since  $c = \min(a + d, b + e)$ , this gives  $c = \min(a, b)$ . Then the maximum exponent is

$$\min(\max(a, b), a + b + e - c, b + a + d - c) = \max(a, b) = a + b - \min(a, b) = a + b - \min(a, b, c)$$

Case 2.1.2) Assume  $c \neq 0$ ,  $f \neq 2$ , but  $a + d = b + e$ . As before, if  $f \mid (P - P')$ , then  $d = e = 0$ . In this case also  $a = b$ . Assume  $c \leq a$ . Note that  $f^{a-c} \mid (P - P')$  and

$\max(a, b) + \min(a + d, b + e) - c = a + b - c$ . Then the maximum exponent is

$$\min(a + b, \max(a, c) + b - c, \max(b, c) + a - c) = a + b - c = a + b - \min(a, b, c).$$

Alternately, assume  $c > a$ . Then for some  $k$ ,  $f^k \parallel (P - P')$ . The maximum exponent is

$$\min(a + b, \max(a, b) + k, b, \max(b, c) + a - c) = b = a + b - a = a + b - \min(a, b, c)$$

Conversely, assume  $f \nmid (P - P')$ . Then  $c \geq a + d = b + e$  and the maximum exponent is

$$\begin{aligned} \min(a + b, \max(a, b), \max(a, c) + b + e - c, \max(b, c) + a + d - c) &= \min(\max(a, b), a + d) \\ &= \max(a, b) = a + b - \min(a, b) = a + b - \min(a, b, c) \end{aligned}$$

Case 2.2.1) Let  $f = 2$ . Assume  $a + d \neq b + e$ . Then  $2^{\min(c, k)} \parallel 2P$  and  $2^{\min(c, k)} \parallel 2P'$ , so that  $2^{\min(c, k)-1} \parallel P, P'$ . If  $c > 1$  and  $k > 1$ , then  $d = e = 0$  and as before the largest exponent is  $a + b - \min(a, b, c)$ . Assume  $c > 1$ ,  $k \leq 1$ . Then  $k = 1$  and  $c = \min(a + d, b + e) - 1$ . The largest exponent is then

$$\min(a + b, \max(a, b) + 1, \max(a, c) + b + e - c, \max(b, c) + a + d - c)$$

If  $a \leq \min(b, c)$  this reduces to  $b + \min(e, 1) = a + b + \min(e, 1) - \min(a, b, c)$ .  $c + 1 \leq a + d \leq c + d$ , so  $d \geq 1$ . Note that if  $e \geq 1$  then this is a special case and  $h = 2$ . If  $e = 0$ , it is the same as before. The cases when  $b \leq \min(a, c)$  are similar.

If  $c \leq \min(a, b)$ , this exponent is  $\min(\max(a, b) + 1, a + b + e - c, b + a + d - c)$ . Without loss of generality, assume  $a + d > b + e$  so that  $c = a + d - 1 \geq c + d - 1$ , so that  $d = 1$  and  $a = c$ . Then the exponent is  $\min(b + 1, b + e, b + d) = b + \min(1, e, d) = a + b + \min(1, e, d) - \min(a, b, c)$ . Note again that if  $e \geq 1$  and  $d \geq 1$ , then this is the special case where  $h = 2$ . Otherwise, it

is the same as before.

Assume  $c = 1$ . Then  $k \geq 1$ . Then the exponent is

$$\min(a + b, \max(a, b) + \min(a + d, b + e) - 1, a + b + e - 1, a + b + d - 1).$$

If  $d = 0$  or  $e = 0$ ,  $h = 1$  and this is  $a + b - 1 = a + b - \min(a, b, c)$ . Otherwise,  $h = 2$  and the exponent is  $a + b = a + b + 1 - \min(a, b, c)$ .

Case 2.2.2) Lastly, assume that  $c \neq 0$  but  $a + d = b + e$ . For some  $k$ ,  $2^k \parallel (P - P')$ .  $c + k \geq a + d$ . If  $c > 1$  and  $k > 1$ , then  $d = e = 0$ ,  $a = b$ . The exponent is then  $\min(2a, a + k, \max(a, c) + b - c)$ . If  $c > a$ , this is  $\min(2a, a + k, a) = a = a + b - \min(a, b, c)$ . If  $c \leq a$ , the exponent is  $\min(2a, a + k, 2a - c) = 2a - c = a + b - \min(a, b, c)$ .

Alternately, if  $c = 1$ , then  $k \geq a + d - 1$  and the exponent is

$$\min(a + b, \max(a, b) + k, a + b + e - 1, b + a + d - 1) = \min(a + b, a + b + e - 1, a + b + d - 1).$$

If  $e > 0$  and  $d > 0$ ,  $h = 2$  and this exponent is  $a + b = a + b + 1 - \min(a, b, c)$ . If  $e = 0$  or  $d = 0$ ,  $h = 1$  and this is  $a + b - 1 = a + b - \min(a, b, c)$ .

If  $k = 1$  then  $c \geq 1$  and specifically  $c \geq a + d - 1 = b + e - 1$ . If  $c \leq \min(a, b)$ , then  $d = e = 1$  so that  $h = 2$ ,  $c = a = b$  and the exponent is

$$\min(2a, a + 1) = a + 1 = a + b + 1 - \min(a, b, c)$$

If  $a \leq \min(b, c)$ , then  $d \geq e$  and the exponent is

$$\min(a + b, b + 1, b + e, \max(b, c) + a + d - c) = \min(b + 1, b + e)$$

If  $e \geq 1$ , then  $d \geq e \geq 1$ , so  $h = 2$  and in this case the exponent is  $b + 1 = a + b + 1 - \min(a, b, c)$ .

If  $e = 0$ ,  $h = 1$  and in this case the exponent is  $b = a + b - \min(a, b, c)$ .

Therefore,

$$L(I \cdot J) = hQQ'/s$$

so that for  $f \neq 2$ , the highest exponent is  $a + b - \min(a, b, c)$  and for  $f = 2$ , the highest exponent is  $a + b - \min(a, b, c)$  if  $h = 1$  and  $a + b + 1 - \min(a, b, c)$  if  $h = 2$ . Note that this is still divisible by  $s$ . After that  $s$  is factored out, the result is  $q = hQQ'/s^2$ .

There are integers  $t$ ,  $u$ , and  $v$  such that  $tQ + uQ' + v(P + P') = s$ . First, consider divisibility by  $s$ . This is trivial for every term except  $N + PP'$ . By the definition of  $s$ ,

$$P + P' \equiv 0 \pmod{s}$$

$$P' \equiv -P \pmod{s}$$

$$PP' \equiv -P^2 \pmod{s}$$

$$N + PP' \equiv N - P^2 \pmod{s}$$

and since  $s \mid Q$  and  $Q \mid (N - P^2)$ ,  $s \mid N + PP'$ .

The linear combination of the last three elements with coefficients  $t$ ,  $u$ , and  $v$  respectively is:

$$s\sqrt{N} + tQP' + uQ'P + v(N + PP')$$

so that it is evident that after  $s$  is factored out, the remaining ideal is primitive. Since this is the element of  $I \cdot J$  with the smallest coefficient of  $\sqrt{N}$ , clearly  $p = t(Q/s)P' + u(Q'/s)P + v(N + PP')/s$ , modulo  $L(I \cdot J)$ . Then,

$$\begin{aligned} p &= t(Q/s)P' + (s - tQ - v(P + P'))P/s + v(N + PP')/s \\ &\equiv P + v(N - P^2)/s \pmod{Q/s} \\ &\equiv P \pmod{Q/s} \end{aligned}$$

since  $Q \mid (N - P^2)$ . By symmetric arguments,  $p \equiv P' \pmod{Q'/s}$ .

To prove (A.24), consider:

$$\begin{aligned}
(P + P')sp &= (P + P')(tQP' + uQ'P + v(N + PP')) \\
&= (P + P')(tQP' + uQ'P) + (P + P')(N + PP')v \\
&= (P + P')(tQP' + uQ'P) + (s - tQ - uQ')(N + PP') \\
&= s(N + PP') + tQ((P')^2 - D) + uQ'(P^2 - N) \\
&\equiv s(N + PP') \pmod{QQ'}
\end{aligned}$$

Therefore,  $(P + P')p \equiv N + PP' \pmod{QQ'/s}$ . **QED**

Observe that when  $h = 1$  (A.21) could be restated as

$$L(I \cdot J) = L(I)L(J)/s^2 \tag{A.26}$$

remembering that  $L(I)$  is defined as the smallest positive rational integer in  $I$ . This equation is proven in [32] and will be useful later.

Also observe that for  $h = 1$ , the equations describing the product of two ideals correspond exactly to the composition of two quadratic forms. Shanks notes this in [28]. Therefore, the equations concerning distance and multiplication of ideals will correspond to distance and composition of quadratic forms.

The case when  $h = 2$  connects composition of quadratic forms of discriminant  $\equiv 1 \pmod{4}$  to multiplication of ideals. If  $F$  and  $G$  are two quadratic forms with discriminant  $N \equiv 1 \pmod{4}$ , then  $2F$  and  $2G$  have discriminant  $4N$  and correspond to ideals  $I_{2F}$  and  $I_{2G}$  in  $\mathbb{Z}[\sqrt{N}]$ . Multiplying,  $h = 2$  and  $I_{2F} \cdot I_{2G} = I_{2(F*G)}$ . Therefore, although this case will not be considered further, it is readily seen that the distance formulas derived from ideals in  $\mathbb{Z}[\sqrt{N}]$  will still correspond to composition of quadratic forms of discriminant  $\equiv 1 \pmod{4}$ .

## 6 Lattices

Consider lattices in  $\mathbb{Q}(\sqrt{N}) \times \mathbb{Q}(\sqrt{N})$ . Define  $\mathbb{L}$  as the set of all lattices in  $\mathbb{Q}(\sqrt{N}) \times \mathbb{Q}(\sqrt{N})$ . Define the map  $M : \mathbb{Q}(\sqrt{N}) \rightarrow \mathbb{Q}(\sqrt{N}) \times \mathbb{Q}(\sqrt{N})$  by

$$M(\xi) = \langle \xi, \bar{\xi} \rangle$$

Multiplication in  $\mathbb{Q}(\sqrt{N}) \times \mathbb{Q}(\sqrt{N})$  is defined componentwise, that is,  $\langle \xi, \xi' \rangle \cdot \langle \zeta, \zeta' \rangle = \langle \xi\zeta, \xi'\zeta' \rangle$ , so that it is clear that  $M$  is homomorphic and one-to-one.

Distance will relate to a concept called a minimum:

**Definition 6** For a vector  $v = \langle v_1, v_2, \dots, v_d \rangle$ , the *normed body* of  $v$ ,  $\mathcal{R}(v)$  is the set

$$\mathcal{R}(v) = \{ \langle x_1, x_2, \dots, x_d \rangle : x_i \in \mathbb{R}, |x_i| < |v_i|, i = 1, 2, \dots, d \}$$

Abusing notation, denote  $\mathcal{R}(\xi) = \mathcal{R}(\langle \xi, \bar{\xi} \rangle)$ .

A number  $\xi$  (or actually the corresponding vector) is a *minimum* of  $\mathcal{L}$  if  $\mathcal{R}(\xi) \cap \mathcal{L} = \{0\}$ , where 0 is the vector  $\langle 0, 0 \rangle$ .

A lattice  $\mathcal{L}$  is *reduced* if  $1 \in \mathcal{L}$  and 1 is a minimum.

For this case with  $d = 2$ , the normed body is a rectangle in  $\mathbb{R}^2$ . Note that for  $\xi \in \mathbb{Q}(\sqrt{N})$  the normed body  $\mathcal{R}(\xi)$  has area equal to four times the absolute value of the norm  $|\mathcal{N}(\xi)|$ .

To avoid unnecessary generality, this investigation will focus specifically on the lattices corresponding to ideals. Specifically, for the primitive ideal  $I = [Q, \sqrt{N} + P]$ , define the associated lattice containing 1 in  $\mathbb{Q}(\sqrt{N})$  as  $\mathcal{L}_I = [1, (\sqrt{N} + P)/Q]$ .

Conversely, to each lattice containing 1 in  $\mathbb{Q}(\sqrt{N})$  there is an associated primitive lattice (which may or may not be an ideal) in  $\mathbb{Z}[\sqrt{N}]$ . Equation (A.17) defined the function  $L$ . In a similar fashion, for a lattice  $\mathcal{L}$ , define

$$L(\mathcal{L}) = \min\{n \in \mathbb{Z}^+ : n\mathcal{L} \subset \mathbb{Z}[\sqrt{N}]\} \quad (\text{A.27})$$

Then if  $L(\mathcal{L})\mathcal{L}$  is an ideal of  $\mathbb{Z}[\sqrt{N}]$  it is the primitive ideal associated to a lattice  $\mathcal{L}$ . Note that if an ideal  $I$  is associated to a lattice  $\mathcal{L}_I$ , then  $L(I) = L(\mathcal{L}_I)$ . Define

$$\Phi_{\mathbb{L},\mathbb{L}}([Q, \sqrt{N} + P]) = [1, (\sqrt{N} + P)/Q]$$

and

$$\Phi_{\mathbb{L},\mathbb{L}}(\mathcal{L}) = L(\mathcal{L})\mathcal{L}$$

Note that for some lattices  $\mathcal{L}$ ,  $\Phi_{\mathbb{L},\mathbb{L}}(\mathcal{L})$  may not actually be an ideal. Lemma 9 provides conditions for it to be an ideal sufficient for this analysis:

**Lemma 9** *Let  $I$  be a primitive ideal and let  $\mathcal{L} = \Phi_{\mathbb{L},\mathbb{L}}(I)$ . If  $\mathcal{L}'$  is a lattice with basis  $\{1, \xi\}$  and for some  $\theta$ ,  $\theta\mathcal{L}' = \mathcal{L}$ , then  $J = \Phi_{\mathbb{L},\mathbb{L}}(\mathcal{L})$  is a primitive ideal and*

$$(L(I)\theta)J = (L(J))I$$

**Proof:** Let  $I = [Q, \sqrt{N} + P]$ . Then  $\mathcal{L} = [1, (\sqrt{N} + P)/Q]$ . The statement that  $\theta\mathcal{L}' = \mathcal{L}$  requires that

$$\theta \begin{bmatrix} 1 \\ \xi \end{bmatrix} = T \begin{bmatrix} 1 \\ (\sqrt{N} + P)/Q \end{bmatrix}$$

where  $T$  is a  $2 \times 2$  matrix with determinant  $\pm 1$ . Multiplying by  $L(I) = L(\mathcal{L}) = Q$  and  $L(J) = L(\mathcal{L}')$ :

$$Q\theta \begin{bmatrix} L(\mathcal{L}') \\ L(\mathcal{L}')\xi \end{bmatrix} = L(\mathcal{L}')T \begin{bmatrix} Q \\ (\sqrt{N} + P) \end{bmatrix}$$

so that  $(L(I)\theta)J = (L(J))I$ . Therefore,  $J$  is an ideal. It is primitive by the definition of  $\Phi_{\mathbb{L}, \mathbb{I}}$ . **QED**

For an example of minima, consider the lattice  $[1, \sqrt{159} - 12]$ .  $\mathcal{R}(1)$  is a square with sides of length 2 centered at the origin and a simple graph demonstrates that 0 is the only point in the lattice and contained in this square. Therefore 1 is a minimum.  $\sqrt{159} - 12$  is also a minimum.  $\mathcal{R}(\sqrt{159} + 12)$  is a narrower and taller rectangle also centered at the origin.

Given two minima, it is important to be able to determine whether or not there is another minimum between them. In vector format, if  $\langle x_1, y_1 \rangle$  and  $\langle x_2, y_2 \rangle$  are minima with  $|x_1| > |x_2|$  and  $|y_1| < |y_2|$ , these two minima are *adjacent* if there does not exist another minima  $\langle x_3, y_3 \rangle$  such that  $|x_2| < |x_3| < |x_1|$  and  $|y_1| < |y_3| < |y_2|$ .

Voronoi developed a method (and a theorem) concerning adjacent minima ([4], [32]).

**Theorem 10** *Let  $\mathcal{L}$  be a lattice with  $\{\xi, \zeta\}$  as a basis, where  $\xi, \zeta \in \mathbb{Q}(\sqrt{N})$  and suppose that  $\zeta > \xi > 0$ . Then  $\zeta$  and  $\xi$  are adjacent minima of  $\mathcal{L}$  if and only if  $|\bar{\xi}| > |\bar{\zeta}|$  and  $\bar{\zeta}\bar{\xi} < 0$ .*

**Proof:** Assume  $\xi$  and  $\zeta$  are adjacent minima. Since they are both minima,  $|\bar{\xi}| > |\bar{\zeta}|$ , or else  $\zeta$  would not be a minima. Also  $0 < \zeta - \xi < \zeta$ . Since  $\zeta$  is a minima, this requires that  $|\bar{\zeta} - \bar{\xi}| > |\bar{\zeta}|$ . If  $\bar{\zeta}$  and  $\bar{\xi}$  had the same sign, this would not be possible. Therefore,  $\bar{\zeta}\bar{\xi} < 0$ .

Conversely, assume that  $|\bar{\xi}| > |\bar{\zeta}|$  and  $\bar{\zeta}\bar{\xi} < 0$ . Assume that  $\xi$  is not a minimum of  $\mathcal{L}$ . Then there exists some  $\omega \in \mathbb{Q}(\sqrt{N})$  such that  $|\omega| < \xi$  and  $|\bar{\omega}| < |\bar{\xi}|$ . Since  $\omega = a\xi + b\zeta$  for some  $a, b \in \mathbb{Z}$ ,  $|a\xi + b\zeta| < \xi$  and  $|a\bar{\xi} + b\bar{\zeta}| < |\bar{\xi}|$ . If  $ab = 0$ , then either  $a = 0$  or  $b = 0$ . If  $a = 0$ , then the second statement contradicts the hypothesis. If  $b = 0$ , then the first statement gives  $\xi < \xi$ , clearly false. However, if  $ab > 0$  then  $|a\xi + b\zeta| > \xi$  and if  $ab < 0$ , then since  $\bar{\zeta}\bar{\xi} < 0$ ,  $|a\bar{\xi} + b\bar{\zeta}| > |\bar{\xi}|$ . Therefore,  $\xi$  must be a minima. By similar reasoning,  $\zeta$  must be a minima.

Concerning adjacency, assume that there is another minima  $\omega$  between  $\xi$  and  $\zeta$ . Since  $\omega = a\xi + b\zeta$  for some  $a, b \in \mathbb{Z}$ ,  $\xi < |a\xi + b\zeta| < \zeta$  and  $|\bar{\zeta}| < |a\bar{\xi} + b\bar{\zeta}| < |\bar{\xi}|$ . Since  $\zeta > \xi > 0$ , the first statement requires that  $b = 0$  and then the second statement simplifies to  $|a| < 1$ ,

requiring that  $a = 0$  and providing a contradiction. Therefore,  $\xi$  and  $\zeta$  are adjacent minima.

**QED**

From the previous example, it is now possible to check that  $\xi = 1$  and  $\zeta = \sqrt{159} + 12$  are indeed adjacent minima.

The idea that will actually connect to continued fractions (and distance) is the search for a sequence of adjacent minima. This sequence is formed by relating different lattices. The following Lemmas are due to Williams [32].

**Lemma 10** *Let  $\mathcal{L}$  and  $\mathcal{L}'$  be reduced lattices. If  $\xi\mathcal{L}' = \mathcal{L}$ , then  $\xi$  is a minimum of  $\mathcal{L}$ .*

**Proof:** Since  $1 \in \mathcal{L}'$ ,  $\xi \in \mathcal{L}$ . If  $\xi$  is not a minimum of  $\mathcal{L}$ , then there exists a  $\zeta \in \mathcal{L}$  such that  $\zeta \neq 0$  and  $|\zeta| < |\xi|$  and  $|\bar{\zeta}| < |\bar{\xi}|$ . Let  $\beta = \zeta/\xi$ , so that  $\beta \in \mathcal{L}'$ .  $|\beta| = |\zeta/\xi| < 1$  and  $|\bar{\beta}| = |\bar{\zeta}/\bar{\xi}| < 1$ , contradicting the fact that  $\mathcal{L}'$  is reduced. Therefore,  $\xi$  is a minimum of  $\mathcal{L}$ .

**QED**

Now consider the converse of this statement. Note that  $\lfloor x \rfloor$  denotes the floor of  $x$ .

**Lemma 11** *Let  $\mathcal{L} = [1, \xi]$ , where 1 and  $\xi$  are adjacent minima of  $\mathcal{L}$  with  $1 > \xi > 0$ . Let  $\mathcal{L}' = (1/\xi)\mathcal{L}$ . Then  $\mathcal{L}'$  is a reduced lattice.*

**Proof:**  $\mathcal{L}' = (1/\xi)[1, \xi] = [1/\xi, 1] = [1/\xi - \lfloor 1/\xi \rfloor, 1]$ , so that  $1 \in \mathcal{L}'$ . It is sufficient to show that 1 and  $\xi' = 1/\xi - \lfloor 1/\xi \rfloor$  are adjacent minima. First, 1 and  $\xi'$  are a basis for  $\mathcal{L}'$  and  $1 > \xi' > 0$ . Since  $0 < \xi < 1$ ,  $\lfloor 1/\xi \rfloor > 1$ . Since  $\bar{\xi} < 0$ ,  $\bar{\xi}' = 1/\bar{\xi} - \lfloor 1/\xi \rfloor < 0 - 1 = -1$ . Thereby satisfying both the requirement that  $\bar{\xi}' \cdot 1 < 0$  and the requirement that  $|\bar{\xi}'| > 1$ . Therefore, by Theorem 10, 1 and  $\xi'$  are adjacent minima of  $\mathcal{L}'$  and thus  $\mathcal{L}'$  is a reduced lattice. **QED**

Actually, these proofs provide a bit more by actually finding the minimum adjacent to 1 in the new lattice. The next Lemma makes use of this minimum [32]:

**Lemma 12** *Let  $\mathcal{L}$ ,  $\mathcal{L}'$ ,  $\xi$ , and  $\xi'$  be as above. Let  $\zeta$  be the minimum adjacent to  $\xi$  other than 1 in  $\mathcal{L}$ . Then  $\zeta = \xi\xi'$ .*

**Proof:**  $\xi\xi' = \xi(1/\xi - \lfloor 1/\xi \rfloor) = 1 - \xi\lfloor 1/\xi \rfloor$ , so that  $[\xi, \xi\xi']$  is a basis for  $\mathcal{L}$ . Since  $1 > \xi' > 0$ ,  $\xi > \xi\xi' > 0$ . Since  $|\bar{\xi}'| > 1$ ,  $|\overline{\xi\xi'}| > |\bar{\xi}|$ . Since  $\bar{\xi}' < 0$ ,  $\overline{\xi \cdot \xi\xi'} = \overline{(\xi)^2\xi'} < 0$ . Therefore, by Theorem 10,  $\xi$  and  $\xi\xi'$  are adjacent minima. Since  $\xi\xi' \neq 1$ ,  $\zeta = \xi\xi'$ . **QED**

Observe that by a similar process, one could find a reduced lattice  $\mathcal{L}'' = 1/\xi'\mathcal{L}'$ , etc. Then  $\mathcal{L}'' = 1/(\xi\xi')\mathcal{L}$ . To generalize, define  $\xi = \xi_1$  and  $\mathcal{L} = \mathcal{L}_1$  and this is a sequence of reduced lattices and their minima. A chain of adjacent minima of  $\mathcal{L}_1$  may be defined by

$$\theta_n = \prod_{i=1}^{n-1} \xi_i \quad (\text{A.28})$$

and then

$$\theta_n \mathcal{L}_n = \mathcal{L}_1 \quad (\text{A.29})$$

Since each  $\mathcal{L}_n$  is a reduced lattice, by Lemma 10 each  $\theta_n$  is a minimum of  $\mathcal{L}_1$ .

Although it is not true in higher dimensions, it is fairly trivial in 2-d that this chain of adjacent minima provides a complete (although infinite) list of the minima with  $x$ -coordinate between 0 and 1.

**Lemma 13** *Let  $\langle \phi, \bar{\phi} \rangle$  be a minimum of a lattice  $\mathcal{L}$ , with  $0 < \phi < 1$ . Then for some  $n$ ,  $\phi = \theta_n$ , where  $\theta_n$  is defined by equation (A.28)*

Define *distance* in terms of this chain of minima by

$$D_{\mathbb{L}}(\mathcal{L}_n, \mathcal{L}_m) = \log(\theta_n/\theta_m) \quad (\text{A.30})$$

It will become readily apparent that the subscript  $\mathbb{L}$  is unnecessary, but it provides clarity for now. Before continuing it is appropriate to provide an example of these concepts. First, as a reference, consider the steps for the continued fraction expansion of  $\sqrt{159} - 12$  and the quadratic form distances  $D_{\mathbb{F}}$  covered to the end of each step:

$$\begin{aligned}
x_1 &= \frac{1}{\sqrt{159}-12} = \frac{\sqrt{159}+12}{15} = 1 + \frac{\sqrt{159}-3}{15}, & D_{\mathbb{F}}(F_0, F_1) &= \log\left(\frac{\sqrt{159}+12}{15}\right) \\
x_2 &= \frac{15}{\sqrt{159}-3} = \frac{\sqrt{159}+3}{10} = 1 + \frac{\sqrt{159}-7}{10}, & D_{\mathbb{F}}(F_0, F_2) &= \log\left(\frac{\sqrt{159}+13}{10}\right) \\
x_3 &= \frac{10}{\sqrt{159}-7} = \frac{\sqrt{159}+7}{11} = 1 + \frac{\sqrt{159}-4}{11}, & D_{\mathbb{F}}(F_0, F_3) &= \log\left(\frac{2\sqrt{159}+25}{11}\right) \\
x_4 &= \frac{11}{\sqrt{159}-4} = \frac{\sqrt{159}+4}{13} = 1 + \frac{\sqrt{159}-9}{13}, & D_{\mathbb{F}}(F_0, F_4) &= \log\left(\frac{3\sqrt{159}+38}{13}\right) \\
x_5 &= \frac{13}{\sqrt{159}-9} = \frac{\sqrt{159}+9}{6} = 3 + \frac{\sqrt{159}-9}{6}, & D_{\mathbb{F}}(F_0, F_5) &= \log\left(\frac{5\sqrt{159}+63}{6}\right).
\end{aligned}$$

The continued fraction corresponds to quadratic forms which correspond to ideals, which are associated with lattices that contain 1. In this case, the lattice associated with  $x_1$  is  $\mathcal{L}_1 = [1, 1/x_1] = [1, \sqrt{159} - 12]$  and  $1/x_1$  is a minimum adjacent to 1 in  $\mathcal{L}_1$ . From here:

$$\begin{aligned}
\mathcal{L}_2 &= \frac{1}{\sqrt{159}-12} \mathcal{L}_1 = \left[\frac{1}{\sqrt{159}-12}, 1\right] = \left[\frac{\sqrt{159}+12}{15}, 1\right] = \left[1, \frac{\sqrt{159}-3}{15}\right] = [1, 1/x_2] \\
\mathcal{L}_3 &= \frac{15}{\sqrt{159}-3} \mathcal{L}_2 = \left[\frac{15}{\sqrt{159}-3}, 1\right] = \left[\frac{\sqrt{159}+3}{10}, 1\right] = \left[1, \frac{\sqrt{159}-7}{10}\right] = [1, 1/x_3] \\
&\dots
\end{aligned}$$

and it is apparent that this same pattern of correspondance will continue, that is

$$\Phi_{\mathbb{T}, \mathbb{L}}(x_n) = \Phi_{\mathbb{T}, \mathbb{I}}(\Phi_{\mathbb{I}, \mathbb{L}}(x_n)) = [1, 1/x_n] = \mathcal{L}_n \quad (\text{A.31})$$

from which it is also apparent that the sequences of lattices will be periodic.

Computing equation (A.28), for example,  $\theta_3 = (\sqrt{159} - 12)(\frac{\sqrt{159}-3}{15}) = 13 - \sqrt{159}$ . With  $\theta_1 = 1$ ,

$$D(\mathcal{L}_1, \mathcal{L}_3) = \log(1/(13 - \sqrt{159})) = \log((\sqrt{159} + 13)/10)$$

It is readily apparent that the definition of distances in lattices corresponds to the definition given for quadratic forms. Note that these distances must still be considered modulo  $R$ , the regulator, since the sequence of lattices is still cyclic.

## 7 The Generalized Distance Formula

Going back to ideals, note that if  $I_1 = L(\mathcal{L}_1)\mathcal{L}_1$  and  $I_n = L(\mathcal{L}_n)\mathcal{L}_n$  is another ideal corresponding to a lattice later in the same sequence, then

$$\begin{aligned}\theta_n \mathcal{L}_n &= \mathcal{L}_1 \\ L(\mathcal{L}_1)L(\mathcal{L}_n)\theta_n \mathcal{L}_n &= L(\mathcal{L}_1)L(\mathcal{L}_n)\mathcal{L}_1 \\ (L(\mathcal{L}_1)\theta_n)I_n &= (L(\mathcal{L}_n)I_1)\end{aligned}\tag{A.32}$$

where once again, the distance (this time between ideals) is given by  $D(I_1, I_n) = -\log(\theta_n)$ . Now, this definition of distance is well and good for reduced ideals, but as of yet, it hasn't been applied it to non-reduced ideals. To relate the definitions of reduced lattices and continued fractions observe that the definition of a reduced continued fraction implies that for a term  $x_i = \frac{\sqrt{N}+P_{i-1}}{Q_i}$ , being reduced equates to

$$\begin{aligned}\frac{\sqrt{N} + P_{i-1}}{Q_i} &> 1 \\ 0 &< \frac{\sqrt{N} - P_{i-1}}{Q_i} < 1\end{aligned}$$

so that it is clear that if  $\mathcal{L}_x = [1, 1/x]$ , then 1 and  $x$  are adjacent minima and the lattice is reduced. The process of dealing with a non-reduced lattice correlates to the process of reducing a continued fraction as demonstrated in the proof of Lemma 5. See [32] for a more general Lemma.

**Lemma 14** *Let  $I$  be any primitive ideal in  $\mathbb{Z}[\sqrt{N}]$ . There exists a reduced ideal  $I_n$  and a  $\theta_n \in I$  such that*

$$(L(I)\theta_n)I_n = (L(I_n))I \quad (\text{A.33})$$

**Proof:**

Let  $I = [Q, \sqrt{N} + P]$ . Then the associated lattice is  $\mathcal{L}_I = [1, \frac{\sqrt{N}+P}{Q}] = [1, \xi_1]$ . If  $I$  is reduced,  $I_n = I$ ,  $u = L(I)$ , and the proof is done. If  $I$  is not reduced, then  $\mathcal{L}_I$  is not reduced. Without loss of generality, assume that  $0 < \xi_1 < 1$  (since otherwise it would just have to be reduced by an integer.). Let  $\mathcal{L}_2 = 1/\xi_1 \mathcal{L}_I = [1/\xi, 1] = [1, 1/\xi - \lfloor 1/\xi - 1/2 \rfloor]$ . Then  $\xi_1 \mathcal{L}_2 = \mathcal{L}_I$ . Continuing in similar manner<sup>11</sup>, by Lemma 5 and the correspondance between lattices and continued fractions for some  $n, \xi_n$  reduced, and thus  $\mathcal{L}_n$  reduced. As in (A.28), set

$$\theta_n = \prod_{i=1}^{n-1} \xi_i$$

so that

$$\theta_n \mathcal{L}_n = \mathcal{L}_I$$

Then  $(L(I)\theta_n)I_n = (L(I_n))I$ . **QED**

Let  $I_1, J_1$  be reduced primitive ideals. Let  $K_1$  be the primitive ideal found by multiplying  $I_1$  and  $J_1$  and removing a factor and let  $s$  be the factor removed, so that  $(s)K_1 = I_1 J_1$ ,  $s \in \mathbb{Z}$ . By Lemma 14 there exists a reduced ideal  $K_j$  and a  $\lambda_j \in K_1$  such that

$$(L(K_1)\lambda_j)K_j = (L(K_j))K_1 \quad (\text{A.34})$$

corresponding to  $D(K_1, K_j) = -\log(\lambda_j)$ .

---

<sup>11</sup>Note that it is irrelevant whether or not the second components of the intermediate lattices are either minima or adjacent to 1. Also note that, as in Lemma 5, the formula would change slightly when the denominators get small.

Let  $I_n \sim I_1$  and  $J_m \sim J_1$  and let  $H_1$  be the primitive ideal found by multiplying  $I_n$  and  $J_m$  and removing a factor and let  $t$  be the factor removed, so that  $(t)H_1 = I_n J_m$ ,  $t \in \mathbb{Z}$ . By Lemma 14 there exists a reduced ideal  $H_k$  and a  $\eta_k \in K_1$  such that

$$(L(H_1)\eta_k)H_k = (L(H_k))H_1 \quad (\text{A.35})$$

corresponding to  $D(H_1, H_k) = -\log(\eta_k)$ .

Also, there exist minima  $\mu_n$  and  $\phi_m$  in the lattices corresponding to  $I_1$  and  $J_1$ , respectively, such that

$$(L(I_1)\mu_n)I_n = (L(I_n))I_1 \quad (\text{A.36})$$

and

$$(L(J_1)\phi_m)J_m = (L(J_m))J_1 \quad (\text{A.37})$$

corresponding to  $D(I_1, I_n) = -\log(\mu_n)$  and  $D(J_1, J_m) = -\log(\phi_m)$ .

By combining (A.26) and (A.34)-(A.37):

$$\begin{aligned} (L(H_k))K_j &= \left( \frac{L(H_k)L(K_j)}{L(K_1)\lambda_j} \right) K_1 \\ &= \left( \frac{L(H_k)L(K_j)}{L(K_1)\lambda_j s} \right) I_1 J_1 \\ &= \left( \frac{L(H_k)L(K_j)L(I_1)L(J_1)\mu_n\phi_m}{L(K_1)\lambda_j s L(I_n)L(J_m)} \right) I_n J_m \\ &= \left( \frac{L(H_k)L(K_j)s\mu_n\phi_m}{\lambda_j L(I_n)L(J_m)} \right) I_n J_m \\ &= \left( \frac{L(H_k)L(K_j)s\mu_n\phi_m t}{\lambda_j L(I_n)L(J_m)} \right) H_1 \\ &= \left( \frac{L(H_k)L(K_j)s\mu_n\phi_m}{\lambda_j t L(H_1)} \right) H_1 \\ &= \left( \frac{L(K_j)s\mu_n\phi_m\eta_k}{\lambda_j t} \right) H_k \end{aligned}$$

Set

$$\psi = \frac{s\mu_n\phi_m\eta_k}{t\lambda_j}$$

and then

$$(L(K_j)\psi)H_k = (L(H_k))K_j \quad (\text{A.38})$$

Since  $K_j$  and  $H_k$  are reduced, by Lemma 10  $\psi$  is a minimum of the lattice  $\mathcal{L}_{K_j}$ , so that for some  $n$ ,  $\psi = \theta_n$ . Therefore,

$$\begin{aligned} D(K_j, H_k) &= -\log(\psi) = -\log(\mu_n) - \log(\phi_m) - \log(\eta_k) + \log(\lambda_j) - \log(s/t) \\ &= D(I_1, I_n) + D(J_1, J_m) + \zeta \end{aligned}$$

where  $\zeta = D(H_1, H_j) - D(K_1, K_j) + \log(t/s)$  will be small compared to  $D(K_j, H_k)$  for  $m, n$  large.

By the correspondance between multiplication of ideals and composition of quadratic forms, this result may be restated in terms of forms:

**Theorem 11** *If  $F_1 \sim F_n$  are equivalent forms and  $G_1 \sim G_m$  are equivalent forms and  $D_{\rho,1}$  is the reduction distance for  $F_1 * G_1$  and  $D_{\rho,2}$  is the reduction distance for  $F_n * G_m$  and  $s$  and  $t$  are the factors cancelled in each respective composition, then*

$$D(F_1 * G_1, F_n * G_m) = D(F_1, F_n) + D(G_1, G_m) + \zeta$$

where  $\zeta = D_{\rho,2} - D_{\rho,1} + \log(t/s)$ .

Example 13 from the Morrison-Brillhart algorithm is in the principal cycle. By Theorem 11 when a form  $F$  is composed with itself, the distance from 1 to  $F$  is roughly doubled,  $d(1, F^2) = 2d(1, F) + \zeta$ . Therefore, the index is roughly doubled, since distance is roughly

proportional to the difference in indices, so that  $F_3 * F_3 = F_6$ , and  $Q_6 = 9$  is the square of  $Q_3 = 3$ .

Since the square of any symmetry point has first coefficient 1, observe that if the distance around some cycle were unrelated to the distance around the principal cycle, then this result would be affected by which symmetry point this distance was referenced from. From Definition 4  $R = D(F_0, F_\pi)$  in the principal cycle. At this point, it is clear that the distance in other cycles must be the same.

**Lemma 15** *Let  $A$  be a primitive amibiguous cycle with a period  $\pi$ . Then,*

$$R = D(F_0, F_\pi)$$

**Proof:** Let  $\{F_i\}$  have period  $\pi$  and let  $F_0$  and  $F_{\pi/2}$  be the two symmetry points of  $A$ . Then  $F_0 * F_0 = 1 = F_{\pi/2} * F_{\pi/2}$ , with  $D_{\rho,1} = D_{\rho,2} = 0$ ,  $s$  and  $t$  the respective first coefficients. Therefore,

$$0 = D(F_0 * F_0, F_{\pi/2} * F_{\pi/2}) = 2D(F_0, F_{\pi/2}) + \log(t/s) = D(F_0, F_\pi)$$

where the 3rd step is obtained from the 2nd by the fact that the product in  $D(F_0, F_{\pi/2})$  includes the last denominator  $t$  and not the first denominator  $s$ .

Therefore,  $D(F_0, F_\pi) = nR$ . Considering composition of  $F_0$  with forms in the principal cycle, clearly  $D(F_0, F_\pi) \leq R$ , so that  $D(F_0, F_\pi) = R$ . **QED**

## 8 Square Forms Factorization (SQUFOF)

It's not certain how much Shanks may have rigorously proven concerning distances, but based on the understanding he had of distance and infrastructure, he was able to develop Square Forms Factorization. A short example will demonstrate and explain the algorithm: let  $N =$

3193. Expanding the continued fraction (principal cycle),  $Q_{10} = 49$ . The quadratic form for this is  $F = 49x^2 + 58xy - 48y^2$ . Since 49 is a perfect square,  $7x^2 + 58xy - 336y^2$ , which reduces with  $D_\rho = 0$  to  $G = 7x^2 + 100xy - 99y^2$  is a quadratic form whose square is  $F$ . Therefore, by Theorem 8,  $G$  is in a class of order 2 or 1, so that  $G$  is an ambiguous form, so that there are two points of symmetry in its cycle. Since by Theorem 11,  $2D(G_s, G) = D(1, F) \pmod{R}$ . So  $D(G_s, G) = D(1, F)/2 \pmod{R/2}$ . Since the two points of symmetry are  $R/2$  away from each other, this means that there is a symmetry point at distance  $D(1, F)/2$  behind  $G$ . Therefore, a point of symmetry may be found by reversing  $G$  and traveling this short distance. Now if the coefficient at this symmetry point is  $\pm 1$ , then there would have been a 7 somewhere before  $F$  in the continued fraction expansion. If the coefficient is 2, then this symmetry point could be composed with  $G$  to find 14 at an earlier point in the principle cycle. Therefore, the symmetry point provides a nontrivial factor for  $N$ . In this case, after 6 steps it provides 31 as a factor of 3193.

The second phase of this algorithm can be made significantly (at least for larger numbers) faster if the quadratic forms from the continued fraction expansion with indices that are powers of 2 are saved. In this example,  $F = F_{10}$ , so that  $G$  is about the  $5^{th}$  form in its cycle<sup>12</sup>. The composition of  $G^{-1}$  with  $F_4$  and  $F_1$  is close and a simultaneous search in both direction from there quickly finds the symmetry point. In this case, it is only necessary to store  $\log_2 k$  forms for  $k$  steps, so that it is more efficient to check each square to see if it works than to check each square root against the previous pseudo-squares to predict whether it will work.

Formally, here is the algorithm for factoring  $N$ :

---

<sup>12</sup>Roughly, since in this case  $5 \approx 6$ .

```

 $Q_0 \leftarrow 1, P_0 \leftarrow \lfloor \sqrt{N} \rfloor, Q_1 \leftarrow N - P_0^2$ 
 $r \leftarrow \lfloor \sqrt{N} \rfloor$ 
while  $Q_i \neq$  perfect square for some  $i$  even
     $b_i \leftarrow \left\lfloor \frac{r+P_{i-1}}{Q_i} \right\rfloor$ 
     $P_i \leftarrow b_i Q_i - P_{i-1}$ 
     $Q_{i+1} \leftarrow Q_{i-1} + b_i(P_{i-1} - P_i)$ 
    if  $i = 2^n$  for some  $n$ 
        Store  $(Q_i, 2 \cdot P_i)$   $F_0 = (\sqrt{Q_i}, 2 \cdot P_{i-1}, \frac{P_{i-1}^2 - N}{Q_i})$ 
Compose  $F_0$  with stored forms according to the
binary representation of  $i/2$  and store result to  $F_0$ .
 $F_0 = (A, B, C)$ 
 $Q_0 \leftarrow |A|, P_0 \leftarrow B/2, Q_1 \leftarrow |C|$ 
 $q_0 \leftarrow Q_1, p_0 \leftarrow P_0, q_1 \leftarrow Q_0$ 
while  $P_i \neq P_{i-1}$  and  $p_i \neq p_{i-1}$ 
    Apply same recursive formulas to  $(Q_0, P_0, Q_1)$  and  $(q_0, p_0, q_1)$ 
If  $P_i = P_{i-1}$ , either  $Q_i$  or  $Q_i/2$  is a nontrivial factor of  $N$ .
If  $p_i = p_{i-1}$ , either  $q_i$  or  $q_i/2$  is a nontrivial factor of  $N$ .

```

Finding a perfect square that provides a factorization is the slowest part of the algorithm, so the number of steps required to obtain this is a good measure of the total runtime. Let  $W$  be the number of forms examined before a square for is found that provides a factorization. In [29], Shanks states that for  $N$  having  $k$  distinct prime factors<sup>13</sup>,

$$E(W) = \ln(8) \frac{2 + \sqrt{2}}{2} \frac{\sqrt[4]{N}}{2^k - 2}.$$

---

<sup>13</sup>Shanks actually let  $N$  have  $k + 1$  distinct prime factors, so the distance looks slightly different but is equivalent.

Shanks did not provide a proof, and it actually wasn't quite right for all  $N$ , but in 2004, Jason Gower completed a runtime analysis and concluded:

**Proposition 3** [8] *Let  $N$  be square-free and have  $k$  distinct odd prime factors. Let  $W$  be the number of forms that SQUFOF must examine before finding a square form that provides a factorization. Then,*

$$E(W) \sim \begin{cases} \frac{2(\sqrt{2}+1)\sqrt[4]{N} \log 2}{2^k-2} & \text{if } N \equiv 1 \pmod{4}, \\ \frac{3(\sqrt{2}+2)\sqrt[4]{N} \log 2}{2(2^k-2)} & \text{if } N \equiv 2 \text{ or } 3 \pmod{4} \end{cases} \quad (\text{A.39})$$

Note that for  $N \equiv 2 \text{ or } 3 \pmod{4}$ , this is equivalent to Shanks estimate.

Therefore, Square Forms Factorization has an expected runtime of  $O(\sqrt[4]{N})$ .

# Appendix B

## Class Group Related Programs

### Number Theory Functions

This first program, NumberTheory.mag, has all of the basic functions needed for composition of quadratic forms and cycling.

```
//NumberTheory.mag

//inv,d = inverse(A,base);
//Calculates modular inverses. Also handles if A,base not relatively prime
inverse := function(A,base)
    d, inv := Xgcd(A,base);
    if d ne 1 then
        A := Floor(A/d);
        _, inv := Xgcd(A,base);
    end if;
    return inv,d;
end function;

//Solves a system of congruence equations: Chinese Remainder Theorem
congruent := function(size,EQ)
    soln := EQ[1];
    prodSoFar := EQ[2];
    for i := 2 to size do
        cong := EQ[2*i-1];
```

```

        modu := Abs(EQ[2*i]);
        inv,d := inverse(prodSoFar,modu);
        if ((soln-cong) mod d) ne 0 then
            return 0, false;
        end if;
        soln := soln+prodSoFar*inv*(cong-soln);
        prodSoFar := prodSoFar*Floor(modu/d);
        soln := soln mod prodSoFar;
    end for;
    return soln,true;
end function;

//Qform := [Q1,P1,Q2,b1], all positive,
//takes one step
cFracStep := function(formA,root)
    formB := [];
    formB[4] := Floor( (root+formA[2])/formA[3]);
    formB[1] := formA[3];
    formB[2] := formB[4]*formA[3]-formA[2];
    formB[3] := formA[1]+(formA[2]-formB[2])*formB[4];
    return formB;
end function;

//Determines if formA is reduced
goodForm := function(formA,root)
    if (root le (formA[2]-1)) or (formA[2] le Abs(root+0.5-formA[3])) then
        return false;
    end if;
    return true;
end function;

//Given that r is the sqrt(square) mod base, squares base and finds
//sqrt(square) mod base congruent to r.
CompSquare := function(r,square,base)
    base := base*base;
    if base eq 1 then
        return r,true;
    end if;
    inv,d := inverse(r,base);

```

```

    if (square mod d) ne 0 then
        return 0,false;
    end if;

    //In this project the only time d will be bigger than 2 is if d|N:
    if 3 le d then
        d;
        return 0,false;
    end if;

    r := Floor(r + inv*square/d);
    if IsOdd(r) then
        r := r - base;
    end if;

    r := Floor(r/2) mod base;
    return r,true;
end function;

//Reverses a quadratic form and takes a step so that the first coefficient is
//unchanged
reverse := function(formA,root)
    saved := formA[3];
    formA[3] := formA[1];
    formA[1] := saved;
    formA := cFracStep(formA,root);
    return formA;
end function;

//reduce A; does not use A[3].
reduce := function(A,root,N)
    b := Floor((root-A[2])/A[1]);
    A[2] := A[2]+b*A[1];
    A[3] := Floor((N-A[2]^2)/A[1]);

    while (not goodForm(A,root)) do
        A := cFracStep(A,root);
    end while;

    return A;
end function;

```

```

//compose F with itself
squareF := function(A,root,N)
    A[2] := A[2]*2;
    D := 4*N;
    d := Gcd(A[2],A[1]);
    A[1] := A[1]/d;
    if d ge 3 then
        "Factor: ", d;
    end if;
    A[2] := CompSquare(A[2],D,A[1]);
    A[1] := A[1]*A[1];
    if IsOdd(A[2]) then
        A[2] := A[2]+A[1];
    end if;
    A[2] := Floor(A[2]/2);

    return reduce(A,root,N);
end function;

Xgcd3 := function(x,y,z)
    d_xy,a,b := Xgcd(x,y);
    d,a_xy,c := Xgcd(d_xy,z);
    a := a*a_xy;
    b := b*a_xy;
    return d,a,b,c;
end function;

//Compose A*B, slightly slower than A*A above
compose := function(A,B,root,N)
    m,u,v,w := Xgcd3(A[1],B[1],A[2]+B[2]); //before 2* [2]'s, so no /2

    A[2] := Floor((A[1]*B[2]*u+B[1]*A[2]*v+(A[2]*B[2]+N)*w)/m);
    A[1] := Floor(A[1]*B[1]/m^2);

    return reduce(A,root,N);
end function;

```

## Program for Testing a Square

FastReturn was the name Shanks gave to the variation that used composition to test a perfect square quickly. This program provides the function that does this.

```
//FastReturn.mag
//returns bool,int: whether or not the form provides a factor, and
//potentially what the factor is

testF := function(A,saved,size,i,pow,r,root,N)

//Tell the user where a square was found.
"Square found at i=", i;
i := Floor(i/2);
size := size-1;
pow := pow/2;

//Reverse, then take the square root
A := reverse(A,root);
A[1] := r;
A := reduce(A,root,N);

//Use composition with quadratic forms saved from the process of getting
//to this square in order to get close to the desired symmetry point.
while size ge 1 do
    if i ge pow then
        i := i-pow;
        A := compose(A,saved[size],root,N);
    end if;
    size := size-1;
    pow := pow/2;
end while;

//Search in both directions for the symmetry point.
B := reverse(A,root);

BP := 1;
AP := 1;
```

```

//If already at a symmetry point, stop.
if A[2] eq B[2] then
  d := Gcd(A[1],N);
  if d ge 2 then
    return true,d;
  else
    return false,1;
  end if;
end if;

//os keeps track of how much the symmetry point was missed by, just for curiosity.
os := 0;
//Cycle through each direction till a symmetry point is found. Usually pretty close.
while (A[2] ne AP) and (B[2] ne BP) do
  AP := A[2];
  BP := B[2];
  A := cFracStep(A,root);
  B := cFracStep(B,root);
  os := os+1;
end while;

"Missed by", os;

//Determine which direction found the symmetry point and determine if it provides
//a factor.
if A[2] eq AP then
  d := Gcd(A[1],N);
  if d ge 2 then
    return true,d;
  else
    return false,1;
  end if;
end if;
"Overshot";
if B[2] eq BP then
  d := Gcd(B[1],N);
  if d ge 2 then
    return true,d;
  else

```

```

        return false,1;
    end if;
end if;

end function;

```

## Square Forms Factorization

This program defines the function SQFactor(N), which uses SQUFOF with Fast Return to factor N.

```

//SQUFOF.mag

load "NumberTheory.mag";
load "FastReturn.mag";

SQFactor := function(N)

//If N is probably prime, quit.
if IsProbablePrime(N) then
    return 0,0;
end if;

tim := Realtime();

//Set the precision to a higher level if necessary. Only affects Sqrt.
AssertAttribute(FldPr, "Precision", Floor(Log(N)/Log(10)));

root := Floor(Sqrt(N));

A := [1,root,N-root^2];

i := 1;

pow := 4;

saved := [];

```

```

size := 1;

fail := 0;

t := false;

lastP := 0;

while not t do
    //Store forms with indices that are powers of 2
    if i eq pow then
        saved[size] := A;
        size := size+1;
        pow := 2*pow;
    end if;

    //Return if a symmetry point is found.
    if lastP eq A[2] then
        d := Gcd(N,A[2]);
        AssertAttribute(FldPr, "Precision", 28);
        return d,Realtime(tim);
    end if;
    lastP := A[2];
    A := cFracStep(A,root);

    if lastP eq A[2] then
        d := Gcd(N,A[2]);
        AssertAttribute(FldPr, "Precision", 28);
        return d,Realtime(tim);
    end if;
    lastP := A[2];
    A := cFracStep(A,root);

    //Every even step, check if the first coefficient is square.
    t,r := IsSquare(A[1]);
    if t then
        t,f := testF(A,saved,size,i,pow,r,root,N);
        if not t then
            fail := fail+1;

```

```

        end if;
    end if;
    i := i+1;

    //If the original symmetry point is found, quit.
    if (A[1] eq 1) or (A[3] eq 1) then
        f := 0;
        t := true;
    end if;

    //If it fails more than 200 times, it's probably better to
    //multiply N by a constant and start over.
    if fail ge 200 then
        t := true;
        f := 0;
    end if;
end while;

AssertAttribute(FldPr, "Precision", 28);
return f,Realtime(tim);

end function;

```

# Appendix C

## Parallel Programs Using MPI

This is a parallel implementation of Square Forms Factorization that uses composition to break the cycle into segments for each processor to search.

```
//parSQUFOF.c

#include "mpi.h"
#include <stdio.h>
#include "NumberTheory.h" //C version of NumberTheory.mag above
#include "/usr/local/include/gmp.h"
#include "FastReturn.h" //C version of FastReturn.mag above
#include "gmpmpi.h" //Provides Send() and Recv() to send a segment from
                    //the master process to the slave processes.

int main(int argc, char *argv[])
{
    //Flags for communications
    const int STOP = 10;
    const int finished = 11;
    const int segSize = 25;
    const int Npass1 = 12;
    const int Npass2 = 13;

    //Administrative variables
    int myrank,i,np,flag,done;
```

```

//These are the large integers, and quadratic forms, used by this algorithm.
mpz_t N,root,firstBack;
struct form Start;
struct form back[segSize];
char *N_str;
int lengthN;
MPI_Status status;

//Initiate communications and obtain basic information.
MPI_Init(&argc, &argv);
MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
MPI_Comm_size(MPI_COMM_WORLD, &np);

//The min of 2 is an idiot check, the max of 32 is to be polite.
if((np < 2) || (np > 32))
{
    MPI_Finalize();
    return 0;
}

done = 0;
mpz_init(N);
//The master process receives N from the user and sends it to the other processors.
if(myrank==0)
{
    printf("N: \n");
    mpz_inp_str(N,stdin,10);
    lengthN = 1+mpz_sizeinbase(N,36);
    N_str = malloc(lengthN*sizeof(char));
    mpz_get_str(N_str,36,N);
    for(i = 1; i<np; i++)
    {
        MPI_Send(&lengthN,1,MPI_INT,i,Npass1,MPI_COMM_WORLD);
        MPI_Send(N_str,lengthN,MPI_CHAR,i,Npass2,MPI_COMM_WORLD);
    }
    free(N_str);
}
else

```

```

{
    MPI_Recv(&lengthN,1,MPI_INT,0,Npass1,MPI_COMM_WORLD,&status);
    N_str = malloc(lengthN*sizeof(char));
    MPI_Recv(N_str,lengthN,MPI_CHAR,0,Npass2,MPI_COMM_WORLD,&status);
    mpz_set_str(N,N_str,36);
    free(N_str);
}

//Calculate sqrt(N) and find the first quadratic form.
mpz_init(root);
mpz_sqrt(root,N);

mpz_init_set_ui(Start.Q0,1);
mpz_init_set(Start.P,root);
mpz_init_set(Start.Q1,N);
mpz_submul(Start.Q1,root,root);
mpz_init(Start.b);

//Prime the pump by stepping forward a few, already checking for squares.
mpz_init(firstBack);
for(i = 0; i<30; i++)
{
    cFracStep(&(Start),root);
    cFracStep(&(Start),root);
    if(isSquare(Start.Q0,firstBack)&&testF(Start,back,Start,0,firstBack,root,N,&done))
    {
        mpz_clear(root);
        mpz_clear(Start.Q0);
        mpz_clear(Start.P);
        mpz_clear(Start.Q1);
        mpz_clear(Start.b);
        mpz_clear(N);
        MPI_Finalize();
        return 0;
    }
}

//Make big jumps by squaring the quadratic forms.
for(i = 0; i<segSize; i++)

```

```

{
    mpz_init_set(back[i].Q0,Start.Q0);
    mpz_init_set(back[i].P,Start.P);
    mpz_init_set(back[i].Q1,Start.Q1);
    mpz_init(back[i].b);
    compose(&Start,Start,Start,root,N);
}

if (myrank ==0) //Master process
{
    int i;
    int found[64];
    MPI_Request waiting[64],stop[64];
    struct segment Next;
    struct form hIncrement,nextHstart;

    //Prepare to recieve from the other processors.
    for(i = 1; i<np; i++)
    {
        waiting[i] = 0;
        MPI_Irecv(&found[i],1,MPI_INT,i,finished,MPI_COMM_WORLD,&waiting[i]);
    }

    //Initiate the required memory.
    mpz_init_set(hIncrement.Q0,back[segSize-1].Q0);
    mpz_init_set(hIncrement.P,back[segSize-1].P);
    mpz_init_set(hIncrement.Q1,back[segSize-1].Q1);
    mpz_init(hIncrement.b);

    mpz_init_set(Next.start.Q0,back[0].Q0);
    mpz_init_set(Next.start.P,back[0].P);
    mpz_init_set(Next.start.Q1,back[0].Q1);
    mpz_init(Next.start.b);

    mpz_init_set(Next.end.Q0,Start.Q0);
    mpz_init_set(Next.end.P,Start.P);
    mpz_init_set(Next.end.Q1,Start.Q1);
    mpz_init(Next.end.b);
}

```

```

mpz_init_set(Next.halfStart.Q0,back[0].Q0);
mpz_init_set(Next.halfStart.P,back[0].P);
mpz_init_set(Next.halfStart.Q1,back[0].Q1);
mpz_init(Next.halfStart.b);

mpz_init_set(nextHstart.Q0,hIncrement.Q0);
mpz_init_set(nextHstart.P,hIncrement.P);
mpz_init_set(nextHstart.Q1,hIncrement.Q1);
mpz_init(nextHstart.b);

//Cycle through the processors, sending them a segment to search if they're ready for it.
i = 1;
while(!done)
{
    if(i>=np)
        i = 1;
    flag = 0;
    MPI_Test(&waiting[i],&flag,&status);
    if(flag)
    {
        if(found[i])
        {
            printf("Something found by rank %d\n",i);
            done = found[i];
        }
        else
        {
            Send(Next,i);
            MPI_Irecv(&found[i],1,MPI_INT,i,finished,MPI_COMM_WORLD,&waiting[i]);
            ftof(&Next.start,Next.end);
            ftof(&Next.halfStart,nextHstart);
            compose(&nextHstart,nextHstart,hIncrement,root,N);
            compose(&Next.end,nextHstart,nextHstart,root,N);
        }
    }
    i++;
}

//When a factor has been found, cycle through to tell everyone.

```

```

    for(i=1; i<np; i++)
        MPI_Isend(&done,1,MPI_INT,i,STOP,MPI_COMM_WORLD,&stop[i]);

    //Clear the memory used.
    mpz_clear(hIncrement.Q0); mpz_clear(hIncrement.P);
    mpz_clear(hIncrement.Q1); mpz_clear(hIncrement.b);
    mpz_clear(Next.start.Q0); mpz_clear(Next.start.P);
    mpz_clear(Next.start.Q1); mpz_clear(Next.start.b);
    mpz_clear(Next.end.Q0); mpz_clear(Next.end.P); mpz_clear(Next.end.Q1);
    mpz_clear(Next.end.b);
    mpz_clear(Next.halfStart.Q0); mpz_clear(Next.halfStart.P);
    mpz_clear(Next.halfStart.Q1); mpz_clear(Next.halfStart.b);
    mpz_clear(nextHstart.Q0); mpz_clear(nextHstart.P);
    mpz_clear(nextHstart.Q1); mpz_clear(nextHstart.b);

    //The master shouldn't quit until all of the other processors have successfully gotten the word.
    for(i=1; i<np; i++)
        MPI_Wait(&stop[i],&status);
}
else
{
    struct segment Mine;
    MPI_Request last;
    MPI_Request ready;
    int index;
    int lineStart;
    int test,found = 0,end = 0;

    //Initiate the required memory.
    mpz_init(Mine.start.Q0); mpz_init(Mine.start.P); mpz_init(Mine.start.Q1);
    mpz_init(Mine.start.b);
    mpz_init(Mine.end.Q0); mpz_init(Mine.end.P); mpz_init(Mine.end.Q1);
    mpz_init(Mine.end.b);
    mpz_init(Mine.halfStart.Q0); mpz_init(Mine.halfStart.P);
    mpz_init(Mine.halfStart.Q1); mpz_init(Mine.halfStart.b);

    //Prepare to receive the message that the job is done.
    MPI_Irecv(&done,1,MPI_INT,0,STOP,MPI_COMM_WORLD,&last);

```

```

//Keep doing this until something is found
while((!flag)&&(!end))
{
    //Output to the screen when ready for a new segment.
    printf("Rank %i, found = %i\n",myrank,found);
    //Alert the master process
    MPI_Isend(&found,1,MPI_INT,0,finished,MPI_COMM_WORLD,&ready);

    Recv(&Mine);

    reduce(&Mine.start,root,N); //Re-calculates the third coefficient

    index = 0;
    lineStart = 0;
    //Step through the quadratic forms until a something is found.
    while(((!isSquare(Mine.start.Q0,firstBack))||
        (!testF(Mine.start,back,Mine.halfStart,index,firstBack,root,N,&test))))&&(!end))
    {

        if(test)
            printf("Rank %i, found a factor\n",myrank);
        cFracStep(&Mine.start,root);
        cFracStep(&Mine.start,root);

        if((!mpz_cmp_ui(Mine.start.Q0,1))||(!mpz_cmp_ui(Mine.start.Q1,1)))
        {
            end = 1;
            found = -1;
            printf("Start found");
        }

        if((!mpz_cmp(Mine.start.Q0,Mine.end.Q0))&&(!mpz_cmp(Mine.start.P,Mine.end.P)))
            end = 1;
        MPI_Test(&last,&flag,&status);
        if(flag)
        {
            printf("Rank %i: STOP recieved.\n",myrank);
            if(done==1)
                end = 1;
        }
    }
}

```

```

        }
        index = index+1;
    }
    if(test)
    {
        found = 1;
        end = 1;
    }
    MPI_Test(&last,&flag,&status);
}

//Output status when done.
printf("found = %i\n",found);
MPI_Isend(&found,1,MPI_INT,0,finished,MPI_COMM_WORLD,&ready);

    mpz_clear(Mine.start.Q0); mpz_clear(Mine.start.P);
mpz_clear(Mine.start.Q1); mpz_clear(Mine.start.b);
    mpz_clear(Mine.end.Q0); mpz_clear(Mine.end.P); mpz_clear(Mine.end.Q1);
mpz_clear(Mine.end.b);
    mpz_clear(Mine.halfStart.Q0); mpz_clear(Mine.halfStart.P);
mpz_clear(Mine.halfStart.Q1); mpz_clear(Mine.halfStart.b);
    printf("Rank %d,done = %d\n",myrank,done);
}

printf("Rank = %d: Done\n",myrank);

MPI_Finalize();

for(i=0; i<segSize; i++)
{
    mpz_clear(back[i].Q0);
    mpz_clear(back[i].P);
    mpz_clear(back[i].Q1);
    mpz_clear(back[i].b);
}

mpz_clear(Start.Q0);
mpz_clear(Start.P);
mpz_clear(Start.Q1);
mpz_clear(Start.b);

```

```

mpz_clear(N);
mpz_clear(root);

return 0;
}

```

## Include file for using GMP with MPI

This file includes the basic operations required to send dynamic integers between two processors. The two functions used by other programs are Send() and Recv() which send and receive segments to be searched.

```

//gmpmpi.h

struct segment {
    struct form start;
    struct form end;
    struct form halfStart; //The quadratic form whose square is start.
};

void Send(struct segment A, int dest);
void Recv(struct segment *A);

//Converts the segment to an array of strings.
//Note that only the first two components of eqch quadratic form are sent.
void form_type(struct segment A, char *A_str[6], int lengths[6])
{
    int i;

    lengths[0] = 1+mpz_sizeinbase(A.start.Q0,36);
    lengths[1] = 1+mpz_sizeinbase(A.start.P,36);
    lengths[2] = 1+mpz_sizeinbase(A.end.Q0,36);
    lengths[3] = 1+mpz_sizeinbase(A.end.P,36);
    lengths[4] = 1+mpz_sizeinbase(A.halfStart.Q0,36);
    lengths[5] = 1+mpz_sizeinbase(A.halfStart.P,36);

    for(i = 0; i<6; i= i+1)

```

```

{
    A_str[i] = malloc(lengths[i]*sizeof(char));
}

mpz_get_str(A_str[0],36,A.start.Q0);
mpz_get_str(A_str[1],36,A.start.P);
mpz_get_str(A_str[2],36,A.end.Q0);
mpz_get_str(A_str[3],36,A.end.P);
mpz_get_str(A_str[4],36,A.halfStart.Q0);
mpz_get_str(A_str[5],36,A.halfStart.P);
}

//Allocates memory on the recieving processor for the incoming segment.
void reform_type(char *A_str[6],int lengths[6])
{
    int i;
    for(i = 0; i<6; i++)
        A_str[i] = malloc(lengths[i]*sizeof(char));
}

//Converts the character string array back into a segment
void unpack(struct segment *A, char *A_str[6])
{
    mpz_set_str((*A).start.Q0,A_str[0],36);
    mpz_set_str((*A).start.P,A_str[1],36);
    mpz_set_str((*A).end.Q0,A_str[2],36);
    mpz_set_str((*A).end.P,A_str[3],36);
    mpz_set_str((*A).halfStart.Q0,A_str[4],36);
    mpz_set_str((*A).halfStart.P,A_str[5],36);
}

//Sends a segment
void Send(struct segment A, int dest)
{
    char *As[6];
    int lengths[6];
    const int tags[6] = {1,2,3,4,5,6};
    int i;

```

```

    form_type(A,As,lengths);
    printf("Sending lengths: %d, %d, %d\n",lengths[0],lengths[1],lengths[2]);

    MPI_Send(&lengths[0],6,MPI_INT,dest,0,MPI_COMM_WORLD);
    for(i = 0; i<6; i++)
        MPI_Send(As[i],lengths[i],MPI_CHAR,dest,tags[i],MPI_COMM_WORLD);

    for(i = 0; i<6; i++)
        free(As[i]);
}

//Recieves a segment
void Recv(struct segment *A)
{
    char *As[6];
    int lengths[6];
    const int tags[6] = {1,2,3,4,5,6};
    int i;
    MPI_Status halt;

    MPI_Recv(&lengths[0],6,MPI_INT,0,0,MPI_COMM_WORLD,&halt);
    printf("Recving lengths: %d, %d, %d\n",lengths[0],lengths[1],lengths[2]);
    reform_type(As,lengths);
    for(i = 0; i<6; i++)
        MPI_Recv(As[i],lengths[i],MPI_CHAR,0,tags[i],MPI_COMM_WORLD,&halt);
    unpack(A,As);
    printf("P = ");
    printf(As[1]);
    printf(" = ");
    mpz_out_str(stdout,10,(*A).start.P);
    printf("\n");
    for(i=0; i<6; i++)
        free(As[i]);
}

```